

# Namespace MB.Common.Aspects

## Classes

### [LoadingDialogAspect](#)

Provides an aspect that displays a loading dialog while a target method is executing.

### [LogAspect](#)

Provides logging functionality for methods using aspect-oriented programming.

# Class LoadingDialogAspect

Namespace: [MB.Common.Aspects](#)

Assembly: MB.Common.dll

Provides an aspect that displays a loading dialog while a target method is executing.

```
[Aspect(Scope.Global)]
[Injection(typeof(LoadingDialogAspect))]
public class LoadingDialogAspect : Attribute, _Attribute
```

## Inheritance

[object](#) ← [Attribute](#) ← LoadingDialogAspect

## Implements

[Attribute](#)

## Inherited Members

[Attribute.GetCustomAttributes\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo\)](#) , [Attribute.GetCustomAttributes\(MemberInfo, bool\)](#) ,  
[Attribute.IsDefined\(MemberInfo, Type\)](#) , [Attribute.IsDefined\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo\)](#) , [Attribute.GetCustomAttributes\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, bool\)](#) , [Attribute.IsDefined\(ParameterInfo, Type\)](#) ,  
[Attribute.IsDefined\(ParameterInfo, Type, bool\)](#) , [Attribute.GetCustomAttribute\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Module, Type\)](#) , [Attribute.GetCustomAttributes\(Module\)](#) ,  
[Attribute.GetCustomAttributes\(Module, bool\)](#) , [Attribute.GetCustomAttributes\(Module, Type, bool\)](#) ,  
[Attribute.IsDefined\(Module, Type\)](#) , [Attribute.IsDefined\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(Module, Type\)](#) , [Attribute.GetCustomAttribute\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttributes\(Assembly\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, bool\)](#) , [Attribute.IsDefined\(Assembly, Type\)](#) ,  
[Attribute.IsDefined\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttribute\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttribute\(Assembly, Type, bool\)](#) , [Attribute.Equals\(object\)](#) ,  
[Attribute.GetHashCode\(\)](#) , [Attribute.Match\(object\)](#) , [Attribute.IsDefaultAttribute\(\)](#) ,

[Attribute.TypeId](#) , [object.ToString\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Remarks

This aspect intercepts method calls and displays a progress dialog using an [IDialogService](#) while the target method is being executed. The dialog is automatically closed when the method completes or throws an exception. If the target method returns a [Task](#), the dialog remains open until the asynchronous operation is completed. This aspect is typically applied to methods where long-running operations occur, to improve user experience by providing visual feedback during execution.

## Constructors

### LoadingDialogAspect()

```
public LoadingDialogAspect()
```

## Fields

### \_\_a\$instance

```
public static readonly LoadingDialogAspect __a$instance
```

#### Field Value

[LoadingDialogAspect](#)

## Methods

### HandleExceptions(Func<object[], object>, object[], string, Type)

Executes the specified method while handling any exceptions that occur during its execution.

```
[Advice(Kind.Around, Targets = Target.Method)]  
public object HandleExceptions(Func<object[], object> method, object[] arguments, string  
methodName, Type declaringType)
```

## Parameters

**method** `Func<object[], object>`

The target method to be executed, represented as a delegate.

**arguments** `object[]`

An array of arguments to be passed to the target method.

**methodName** `string`

The name of the method being executed.

**declaringType** `Type`

The type that declares the method being executed.

## Returns

`object`

The result of the executed method.

## Remarks

If an exception is thrown during the execution of the target method, the method ensures that any active dialog managed by the `_dialogService` is closed before rethrowing the exception.

## OnEntry()

Executes logic before the target method is invoked.

```
[Advice(Kind.Before, Targets = Target.Method)]
public void OnEntry()
```

## Remarks

This method is typically used to perform pre-execution tasks, such as initializing services or displaying UI elements. It is invoked automatically by the aspect framework.

## OnExit()

Executes after the target method completes, ensuring any associated dialog is closed.

```
[Advice(Kind.After, Targets = Target.Method)]  
public void OnExit()
```

## Remarks

This method is invoked automatically as an aspect after the execution of the target method. If a dialog service is available, it ensures that the dialog is closed.

# Class LogAspect

Namespace: [MB.Common.Aspects](#)

Assembly: MB.Common.dll

Provides logging functionality for methods using aspect-oriented programming.

```
[Aspect(Scope.Global)]
[Injection(typeof(LogAspect))]
public class LogAspect : Attribute, _Attribute
```

## Inheritance

[object](#) ← [Attribute](#) ← LogAspect

## Implements

[Attribute](#)

## Inherited Members

[Attribute.GetCustomAttributes\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo\)](#) , [Attribute.GetCustomAttributes\(MemberInfo, bool\)](#) ,  
[Attribute.IsDefined\(MemberInfo, Type\)](#) , [Attribute.IsDefined\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo\)](#) , [Attribute.GetCustomAttributes\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, bool\)](#) , [Attribute.IsDefined\(ParameterInfo, Type\)](#) ,  
[Attribute.IsDefined\(ParameterInfo, Type, bool\)](#) , [Attribute.GetCustomAttribute\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Module, Type\)](#) , [Attribute.GetCustomAttributes\(Module\)](#) ,  
[Attribute.GetCustomAttributes\(Module, bool\)](#) , [Attribute.GetCustomAttributes\(Module, Type, bool\)](#) ,  
[Attribute.IsDefined\(Module, Type\)](#) , [Attribute.IsDefined\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(Module, Type\)](#) , [Attribute.GetCustomAttribute\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttributes\(Assembly\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, bool\)](#) , [Attribute.IsDefined\(Assembly, Type\)](#) ,  
[Attribute.IsDefined\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttribute\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttribute\(Assembly, Type, bool\)](#) , [Attribute.Equals\(object\)](#) ,  
[Attribute.GetHashCode\(\)](#) , [Attribute.Match\(object\)](#) , [Attribute.IsDefaultAttribute\(\)](#) ,

[Attribute.TypeId](#) , [object.ToString\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Remarks

This attribute can be applied to methods to automatically log method entry, exit, and exceptions. It uses aspect-oriented programming to inject logging behavior at runtime.

## Constructors

### LogAspect()

```
public LogAspect()
```

## Fields

### a\$instance

```
public static readonly LogAspect __a$instance
```

#### Field Value

[LogAspect](#)

## Methods

### HandleExceptions(Func<object[], object>, object[], string, Type)

Executes the specified method and handles any exceptions that occur during its execution.

```
[Advice(Kind.Around, Targets = Target.Method)]
public object HandleExceptions(Func<object[], object> method, object[] arguments, string
methodName, Type declaringType)
```

#### Parameters

method [Func<object\[\], object>](#)

The delegate representing the target method to be executed.

#### arguments [object](#)[]

An array of arguments to be passed to the target method.

#### methodName [string](#)

The name of the target method being executed.

#### declaringType [Type](#)

The type that declares the target method.

### Returns

#### [object](#)

The result of the target method execution.

### Remarks

This method wraps the execution of a target method, logging any exceptions that are thrown. If an exception occurs, it is logged with details about the method name, declaring type, and the exception message, and then rethrown for further handling by the caller.

## OnEntry(string, Type)

Logs an informational message when entering a method.

```
[Advice(Kind.Before, Targets = Target.Method)]
public void OnEntry(string methodName, Type declaringType)
```

### Parameters

#### methodName [string](#)

The name of the method being entered.

#### declaringType [Type](#)

The type that declares the method being entered.

## Remarks

This method is intended to be used as an aspect to log method entry points. It captures the method name and declaring type and logs them along with the current timestamp.

## OnExit(string, Type)

Logs information when a method exits, including the method name, declaring type, and timestamp.

```
[Advice(Kind.After, Targets = Target.Method)]  
public void OnExit(string methodName, Type declaringType)
```

### Parameters

**methodName** [string](#)

The name of the method that is exiting.

**declaringType** [Type](#)

The type that declares the method being exited.

## Remarks

This method is intended to be used as an aspect to provide logging functionality for method exit events. It captures the method name and declaring type dynamically and logs the information along with the current timestamp.

# Namespace MB.Common.Attributes

## Classes

### [ErrorInformationAttribute](#)

An attribute used to annotate fields with additional error information. This is useful for attaching meaningful error messages or metadata to enum fields or other constants.

### [GuidAttribute](#)

Represents a custom attribute that can be applied to fields to associate them with a specific GUID.

### [RequiredSharedParametersAttribute](#)

Represents an attribute to specify required shared parameters for a given target.

### [RevitCategoriesAttribute](#)

Represents an attribute to associate a target with a set of Revit built-in categories.

### [SharedParameterTypeAttribute](#)

Represents an attribute to specify the type of a shared parameter.

# Class ErrorInformationAttribute

Namespace: [MB.Common.Attributes](#)

Assembly: MB.Common.dll

An attribute used to annotate fields with additional error information. This is useful for attaching meaningful error messages or metadata to enum fields or other constants.

```
[AttributeUsage(AttributeTargets.Field)]
public class ErrorInformationAttribute : Attribute, _Attribute
```

## Inheritance

[object](#) ← [Attribute](#) ← ErrorInformationAttribute

## Implements

[Attribute](#)

## Inherited Members

[Attribute.GetCustomAttributes\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo\)](#) , [Attribute.GetCustomAttributes\(MemberInfo, bool\)](#) ,  
[Attribute.IsDefined\(MemberInfo, Type\)](#) , [Attribute.IsDefined\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo\)](#) , [Attribute.GetCustomAttributes\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, bool\)](#) , [Attribute.IsDefined\(ParameterInfo, Type\)](#) ,  
[Attribute.IsDefined\(ParameterInfo, Type, bool\)](#) , [Attribute.GetCustomAttribute\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Module, Type\)](#) , [Attribute.GetCustomAttributes\(Module\)](#) ,  
[Attribute.GetCustomAttributes\(Module, bool\)](#) , [Attribute.GetCustomAttributes\(Module, Type, bool\)](#) ,  
[Attribute.IsDefined\(Module, Type\)](#) , [Attribute.IsDefined\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(Module, Type\)](#) , [Attribute.GetCustomAttribute\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttributes\(Assembly\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, bool\)](#) , [Attribute.IsDefined\(Assembly, Type\)](#) ,  
[Attribute.IsDefined\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttribute\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttribute\(Assembly, Type, bool\)](#) , [Attribute.Equals\(object\)](#) ,  
[Attribute.GetHashCode\(\)](#) , [Attribute.Match\(object\)](#) , [Attribute.IsDefaultAttribute\(\)](#) ,

[Attribute.TypeId](#) , [object.ToString\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Constructors

### ErrorInformationAttribute(string)

Initializes a new instance of the [ErrorInformationAttribute](#) class with the specified error information string.

```
public ErrorInformationAttribute(string errorInformation)
```

#### Parameters

**errorInformation** [string](#)

The error information to associate with the field.

## Properties

### ErrorInformation

Gets the error information message associated with the field.

```
public string ErrorInformation { get; }
```

#### Property Value

[string](#)

# Class GuidAttribute

Namespace: [MB.Common.Attributes](#)

Assembly: MB.Common.dll

Represents a custom attribute that can be applied to fields to associate them with a specific GUID.

```
[AttributeUsage(AttributeTargets.Field)]
public class GuidAttribute : Attribute, _Attribute
```

## Inheritance

[object](#) ← [Attribute](#) ← GuidAttribute

## Implements

[Attribute](#)

## Inherited Members

[Attribute.GetCustomAttributes\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo\)](#) , [Attribute.GetCustomAttributes\(MemberInfo, bool\)](#) ,  
[Attribute.IsDefined\(MemberInfo, Type\)](#) , [Attribute.IsDefined\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo\)](#) , [Attribute.GetCustomAttributes\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, bool\)](#) , [Attribute.IsDefined\(ParameterInfo, Type\)](#) ,  
[Attribute.IsDefined\(ParameterInfo, Type, bool\)](#) , [Attribute.GetCustomAttribute\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Module, Type\)](#) , [Attribute.GetCustomAttributes\(Module\)](#) ,  
[Attribute.GetCustomAttributes\(Module, bool\)](#) , [Attribute.GetCustomAttributes\(Module, Type, bool\)](#) ,  
[Attribute.IsDefined\(Module, Type\)](#) , [Attribute.IsDefined\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(Module, Type\)](#) , [Attribute.GetCustomAttribute\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttributes\(Assembly\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, bool\)](#) , [Attribute.IsDefined\(Assembly, Type\)](#) ,  
[Attribute.IsDefined\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttribute\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttribute\(Assembly, Type, bool\)](#) , [Attribute.Equals\(object\)](#) ,  
[Attribute.GetHashCode\(\)](#) , [Attribute.Match\(object\)](#) , [Attribute.IsDefaultAttribute\(\)](#) ,  
[Attribute.TypeId](#) , [object.ToString\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Remarks

This attribute allows developers to tag fields with a GUID value, which can then be used for identification, serialization, or other purposes where a globally unique identifier is required.

## Constructors

### GuidAttribute(string)

Initializes a new instance of the [GuidAttribute](#) class with the specified GUID string.

```
public GuidAttribute(string guid)
```

#### Parameters

**guid** [string](#)

The GUID string to associate with the field. Must be in a valid GUID format.

#### Exceptions

[FormatException](#)

Thrown when the provided string is not in a valid GUID format.

## Properties

### Guid

Gets the GUID associated with the field to which this attribute is applied.

```
public string Guid { get; }
```

#### Property Value

[string](#)

# Class RequiredSharedParametersAttribute

Namespace: [MB.Common.Attributes](#)

Assembly: MB.Common.dll

Represents an attribute to specify required shared parameters for a given target.

```
[AttributeUsage(AttributeTargets.Field)]
public class RequiredSharedParametersAttribute : Attribute, _Attribute
```

## Inheritance

[object](#) ← [Attribute](#) ← RequiredSharedParametersAttribute

## Implements

[Attribute](#)

## Inherited Members

[Attribute.GetCustomAttributes\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo\)](#) , [Attribute.GetCustomAttributes\(MemberInfo, bool\)](#) ,  
[Attribute.IsDefined\(MemberInfo, Type\)](#) , [Attribute.IsDefined\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo\)](#) , [Attribute.GetCustomAttributes\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, bool\)](#) , [Attribute.IsDefined\(ParameterInfo, Type\)](#) ,  
[Attribute.IsDefined\(ParameterInfo, Type, bool\)](#) , [Attribute.GetCustomAttribute\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Module, Type\)](#) , [Attribute.GetCustomAttributes\(Module\)](#) ,  
[Attribute.GetCustomAttributes\(Module, bool\)](#) , [Attribute.GetCustomAttributes\(Module, Type, bool\)](#) ,  
[Attribute.IsDefined\(Module, Type\)](#) , [Attribute.IsDefined\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(Module, Type\)](#) , [Attribute.GetCustomAttribute\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttributes\(Assembly\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, bool\)](#) , [Attribute.IsDefined\(Assembly, Type\)](#) ,  
[Attribute.IsDefined\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttribute\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttribute\(Assembly, Type, bool\)](#) , [Attribute.Equals\(object\)](#) ,  
[Attribute.GetHashCode\(\)](#) , [Attribute.Match\(object\)](#) , [Attribute.IsDefaultAttribute\(\)](#) ,  
[Attribute.TypeId](#) , [object.ToString\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Remarks

This attribute allows developers to declare a set of shared parameters that are required for the functionality or configuration of a specific target (e.g., class, method, or property).

## Constructors

### RequiredSharedParametersAttribute(params ToolsSharedParameter[])

Initializes a new instance of the [RequiredSharedParametersAttribute](#) class.

```
public RequiredSharedParametersAttribute(params ToolsSharedParameter[] sharedParameters)
```

#### Parameters

##### sharedParameters [ToolsSharedParameter\[\]](#)

A variable-length array of [ToolsSharedParameter](#) objects that represent the required shared parameters.

#### Exceptions

##### ArgumentNullException

Thrown if the provided `sharedParameters` array is null.

## Properties

### SharedParameters

Gets the array of shared parameters that are required.

```
public ToolsSharedParameter[] SharedParameters { get; }
```

#### Property Value

##### [ToolsSharedParameter\[\]](#)

# Class RevitCategoriesAttribute

Namespace: [MB.Common.Attributes](#)

Assembly: MB.Common.dll

Represents an attribute to associate a target with a set of Revit built-in categories.

```
[AttributeUsage(AttributeTargets.Field)]
public class RevitCategoriesAttribute : Attribute, _Attribute
```

## Inheritance

[object](#) ← [Attribute](#) ← RevitCategoriesAttribute

## Implements

[Attribute](#)

## Inherited Members

[Attribute.GetCustomAttributes\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo\)](#) , [Attribute.GetCustomAttributes\(MemberInfo, bool\)](#) ,  
[Attribute.IsDefined\(MemberInfo, Type\)](#) , [Attribute.IsDefined\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo\)](#) , [Attribute.GetCustomAttributes\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, bool\)](#) , [Attribute.IsDefined\(ParameterInfo, Type\)](#) ,  
[Attribute.IsDefined\(ParameterInfo, Type, bool\)](#) , [Attribute.GetCustomAttribute\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Module, Type\)](#) , [Attribute.GetCustomAttributes\(Module\)](#) ,  
[Attribute.GetCustomAttributes\(Module, bool\)](#) , [Attribute.GetCustomAttributes\(Module, Type, bool\)](#) ,  
[Attribute.IsDefined\(Module, Type\)](#) , [Attribute.IsDefined\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(Module, Type\)](#) , [Attribute.GetCustomAttribute\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttributes\(Assembly\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, bool\)](#) , [Attribute.IsDefined\(Assembly, Type\)](#) ,  
[Attribute.IsDefined\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttribute\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttribute\(Assembly, Type, bool\)](#) , [Attribute.Equals\(object\)](#) ,  
[Attribute.GetHashCode\(\)](#) , [Attribute.Match\(object\)](#) , [Attribute.IsDefaultAttribute\(\)](#) ,  
[Attribute.TypeId](#) , [object.ToString\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Remarks

This attribute can be used to declare which Revit built-in categories are relevant for a specific class, method, or property. It allows for filtering or categorization based on Revit's predefined categories.

## Constructors

### RevitCategoriesAttribute(params BuiltInCategory[])

Initializes a new instance of the [RevitCategoriesAttribute](#) class.

```
public RevitCategoriesAttribute(params BuiltInCategory[] categories)
```

#### Parameters

**categories** BuiltInCategory[]

A variable-length array of BuiltInCategory enums that represent the relevant Revit categories.

#### Exceptions

ArgumentNullException

Thrown if the provided **categories** array is null.

## Properties

### Categories

Gets the array of Revit built-in categories associated with the target.

```
public BuiltInCategory[] Categories { get; }
```

#### Property Value

BuiltInCategory[]

# Class SharedParameterTypeAttribute

Namespace: [MB.Common.Attributes](#)

Assembly: MB.Common.dll

Represents an attribute to specify the type of a shared parameter.

```
[AttributeUsage(AttributeTargets.Class|AttributeTargets.Property|AttributeTargets.Field)]
public class SharedParameterTypeAttribute : Attribute, _Attribute
```

## Inheritance

[object](#) ← [Attribute](#) ← SharedParameterTypeAttribute

## Implements

[Attribute](#)

## Inherited Members

[Attribute.GetCustomAttributes\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo\)](#) , [Attribute.GetCustomAttributes\(MemberInfo, bool\)](#) ,  
[Attribute.IsDefined\(MemberInfo, Type\)](#) , [Attribute.IsDefined\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo\)](#) , [Attribute.GetCustomAttributes\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, bool\)](#) , [Attribute.IsDefined\(ParameterInfo, Type\)](#) ,  
[Attribute.IsDefined\(ParameterInfo, Type, bool\)](#) , [Attribute.GetCustomAttribute\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Module, Type\)](#) , [Attribute.GetCustomAttributes\(Module\)](#) ,  
[Attribute.GetCustomAttributes\(Module, bool\)](#) , [Attribute.GetCustomAttributes\(Module, Type, bool\)](#) ,  
[Attribute.IsDefined\(Module, Type\)](#) , [Attribute.IsDefined\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(Module, Type\)](#) , [Attribute.GetCustomAttribute\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttributes\(Assembly\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, bool\)](#) , [Attribute.IsDefined\(Assembly, Type\)](#) ,  
[Attribute.IsDefined\(Assembly, Type, bool\)](#) , [Attribute.GetCustomAttribute\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttribute\(Assembly, Type, bool\)](#) , [Attribute.Equals\(object\)](#) ,  
[Attribute.GetHashCode\(\)](#) , [Attribute.Match\(object\)](#) , [Attribute.IsDefaultAttribute\(\)](#) ,  
[Attribute.TypeId](#) , [object.ToString\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Remarks

This attribute is used to associate a target (e.g., a class, method, or property) with a specific shared parameter type. It helps define the type of shared parameter in a structured and reusable way.

## Constructors

### SharedParameterTypeAttribute(SharedParameterType)

Initializes a new instance of the [SharedParameterTypeAttribute](#) class.

```
public SharedParameterTypeAttribute(SharedParameterType type)
```

#### Parameters

**type** [SharedParameterType](#)

The [SharedParameterType](#) to associate with the target.

#### Exceptions

ArgumentNullException

Thrown if the provided **type** is null (only if SharedParameterType is a reference type).

## Properties

### Type

Gets the type of the shared parameter associated with this attribute.

```
public SharedParameterType Type { get; }
```

### Property Value

[SharedParameterType](#)

# Namespace MB.Common.Comparers

## Classes

### [BoundingBoxComparer](#)

A utility class for comparing and resolving bounding box coordinates from multiple points.

# Class BoundingBoxComparer

Namespace: [MB.Common.Comparers](#)

Assembly: MB.Common.dll

A utility class for comparing and resolving bounding box coordinates from multiple points.

```
public class BoundingBoxComparer
```

## Inheritance

[object](#) ← BoundingBoxComparer

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### BoundingBoxComparer()

Initializes a new instance of the [BoundingBoxComparer](#) class. Sets initial values for bounds: maximum values for minimum bounds and minimum values for maximum bounds.

```
public BoundingBoxComparer()
```

## Methods

### Compare(XYZ)

Compares the given point with the current bounds and updates the bounds accordingly.

```
public void Compare(XYZ point)
```

## Parameters

**point** XYZ

The 3D point to compare.

## Resolve()

Resolves the current bounds into a minimum and maximum point, extending them by a fixed tolerance.

```
public (XYZ min, XYZ max) Resolve()
```

Returns

(XYZ [Perpendicular](#), XYZ [Normal](#))

A tuple containing:

- The minimum point (Autodesk.Revit.DB.XYZ) of the bounding box.
- The maximum point (Autodesk.Revit.DB.XYZ) of the bounding box.

# Namespace MB.Common.Constants

## Classes

### [FilePaths](#)

Contains static file paths and predefined constants used throughout the application. Paths are based on the user's local application data folder.

### [MbErrorMessages](#)

Contains predefined error message constants used throughout the application. Helps maintain consistency and manageability of user-facing messages.

### [MbfFiles](#)

Provides constants and utility properties for MBF-related file paths.

### [MbfGroups](#)

Provides constant values for MBF parameter groups.

# Class FilePaths

Namespace: [MB.Common.Constants](#)

Assembly: MB.Common.dll

Contains static file paths and predefined constants used throughout the application. Paths are based on the user's local application data folder.

```
public static class FilePaths
```

## Inheritance

[object](#) ← FilePaths

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Fields

## Elbows

Constant for identifying elbow elements.

```
public const string Elbows = "Elbows"
```

## Field Value

[string](#)

## FlexPipes

Constant for identifying flexible pipe elements.

```
public const string FlexPipes = "Flex Pipes"
```

## Field Value

[string](#)

## HazardCoverageJson

Constant for identifying pipe elements.

```
public const string HazardCoverageJson = "HazardCoverage.json"
```

### Field Value

[string](#)

## LogBaseDirectory

Gets the base directory path where log files are stored.

```
public static readonly string LogBaseDirectory
```

### Field Value

[string](#)

### Remarks

The directory path is constructed by combining the local application data folder, a subdirectory named "MicroBIM", and a "Logs" folder. This path is intended to provide a consistent location for storing application log files.

## Others

Constant for identifying miscellaneous or other elements.

```
public const string Others = "Others"
```

### Field Value

[string](#)

## Outlets

Constant for identifying outlet elements.

```
public const string Outlets = "Outlets"
```

Field Value

[string](#)

## PipeAccessories

Constant for identifying pipe accessory elements.

```
public const string PipeAccessories = "Pipe Accessories"
```

Field Value

[string](#)

## Pipes

Constant for identifying pipe elements.

```
public const string Pipes = "Pipes"
```

Field Value

[string](#)

## Sprinklers

Constant for identifying sprinkler elements.

```
public const string Sprinklers = "Sprinklers"
```

Field Value

[string](#)

## Tees

Constant for identifying tee elements.

```
public const string Tees = "Tees"
```

Field Value

[string](#)

## Properties

### AutoSprinklerSpacing

Gets the path to the directory containing Auto Sprinkler Spacing JSON files.

```
public static string AutoSprinklerSpacing { get; }
```

Property Value

[string](#)

### ListingSettingsJsons

Gets the path to the directory containing listing-specific Mapper JSON settings.

```
public static string ListingSettingsJsons { get; }
```

Property Value

[string](#)

### MapperJsons

Gets the path to the directory containing Mapper JSON files.

```
public static string MapperJsons { get; }
```

Property Value

[string](#)

## MicroBimListingSettingsJsons

Gets the path to the directory containing MicroBIM Listing Settings JSON files.

```
public static string MicroBimListingSettingsJsons { get; }
```

Property Value

[string](#)

## SettingsJsons

Gets the path to the directory containing general Mapper JSON settings.

```
public static string SettingsJsons { get; }
```

Property Value

[string](#)

## UserListing

Gets the path to the directory containing user-specific listing settings JSON files.

```
public static string UserListing { get; }
```

Property Value

[string](#)

## UserValuesToStandardJson

Path for mapping user-defined values to a standard JSON format within listing settings.

```
public static string UserValuesToStandardJson { get; }
```

Property Value

[string](#) ↗

# Class MbErrorMessages

Namespace: [MB.Common.Constants](#)

Assembly: MB.Common.dll

Contains predefined error message constants used throughout the application. Helps maintain consistency and manageability of user-facing messages.

```
public static class MbErrorMessages
```

## Inheritance

[object](#) ← MbErrorMessages

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Fields

### MoreThanOneSpaceSelected

Error message displayed when more than one space is selected.

```
public const string MoreThanOneSpaceSelected = "Please select only one space."
```

## Field Value

[string](#)

### NoSpaceSelected

Error message displayed when no space is selected.

```
public const string NoSpaceSelected = "Please select the space."
```

## Field Value

[string](#)

## NoSprinklerSelected

Error message displayed when no sprinkler is selected.

```
public const string NoSprinklerSelected = "Please select at least one sprinkler."
```

### Field Value

[string](#)

## SprinklerNotInsideSpace

Error message displayed when a sprinkler is not located within the selected space.

```
public const string SprinklerNotInsideSpace = "Sprinkler is not inside the selected space."
```

### Field Value

[string](#)

# Class Mbffiles

Namespace: [MB.Common.Constants](#)

Assembly: MB.Common.dll

Provides constants and utility properties for MBF-related file paths.

```
public static class Mbffiles
```

## Inheritance

[object](#) ← Mbffiles

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Properties

## AddinDirectory

Gets the directory of the currently executing assembly (Add-in directory).

```
public static string AddinDirectory { get; }
```

## Property Value

[string](#)

## Remarks

This property dynamically resolves the directory path where the executing assembly is located.

## ToolsSharedParameter

Gets the full path to the shared parameter file (3dfirespf.txt) used by MBF.

```
public static string ToolsSharedParameter { get; }
```

## Property Value

[string](#) ↗

# Class Mbfgroups

Namespace: [MB.Common.Constants](#)

Assembly: MB.Common.dll

Provides constant values for MBF parameter groups.

```
public static class Mbfgroups
```

## Inheritance

[object](#) ← Mbfgroups

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class contains group names for organizing parameters used in the MBF suite. These constants can be used to ensure consistent naming throughout the application.

## Fields

### Fittings

Represents the group name for pipe accessory fittings parameters.

```
public const string Fittings = "MBFire-Pipes-Accessory-Fittings-Parameters"
```

### Field Value

[string](#)

# Namespace MB.Common.Errors

## Classes

### [MbErrorExtensions](#)

Provides extension methods for the [MbError](#) enumeration.

## Enums

### [MbError](#)

Defines a list of error codes used throughout the application. Each enum value is annotated with [Error InformationAttribute](#) to provide a user-friendly error message.

# Enum MbError

Namespace: [MB.Common.Errors](#)

Assembly: MB.Common.dll

Defines a list of error codes used throughout the application. Each enum value is annotated with [ErrorInformationAttribute](#) to provide a user-friendly error message.

```
public enum MbError
```

## Extension Methods

[MbErrorExtensions.GetErrorMessage\(MbError\)](#)

## Fields

[[ErrorInformation\("Please select only one space."\)](#)] MultipleSpacesSelected = 2

Error when more than one space is selected, but only one is expected.

[[ErrorInformation\("Please select the space."\)](#)] NoSpaceSelected = 1

Error when no space is selected by the user.

[[ErrorInformation\("Please select at least one sprinkler."\)](#)] NoSprinklerSelected = 3

Error when no sprinkler is selected.

[[ErrorInformation\("Sprinkler is not inside the selected space."\)](#)] SprinklersNotInsideSpace = 4

Error when selected sprinklers are not located inside the selected space.

# Class MbErrorExtensions

Namespace: [MB.Common.Errors](#)

Assembly: MB.Common.dll

Provides extension methods for the [MbError](#) enumeration.

```
public static class MbErrorExtensions
```

## Inheritance

[object](#) ← MbErrorExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## GetErrorMessage(MbError)

Retrieves the error information associated with the specified [MbError](#).

```
public static string GetErrorMessage(this MbError error)
```

### Parameters

**error** [MbError](#)

The [MbError](#) instance for which the error message is retrieved.

### Returns

[string](#)

# Namespace MB.Common.Errors.Exceptions

## Classes

### [MbException](#)

Represents a custom exception type used within the MicroBIM system. Wraps an [MbError](#) and provides a description derived from it.

# Class MbException

Namespace: [MB.Common.Errors.Exceptions](#)

Assembly: MB.Common.dll

Represents a custom exception type used within the MicroBIM system. Wraps an [MbError](#) and provides a description derived from it.

```
public class MbException : Exception, ISerializable, _Exception
```

## Inheritance

[object](#) ← [Exception](#) ← MbException

## Implements

[ISerializable](#), [Exception](#)

## Inherited Members

[Exception.GetBaseException\(\)](#), [Exception.ToString\(\)](#),  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#), [Exception.GetType\(\)](#),  
[Exception.Message](#), [Exception.Data](#), [Exception.InnerException](#), [Exception.TargetSite](#),  
[Exception.StackTrace](#), [Exception.HelpLink](#), [Exception.Source](#), [Exception.HResult](#),  
[Exception.SerializeObjectState](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.MemberwiseClone\(\)](#)

# Constructors

## MbException(MbError)

Represents a custom exception type used within the MicroBIM system. Wraps an [MbError](#) and provides a description derived from it.

```
public MbException(MbError mbError)
```

## Parameters

mbError [MbError](#)

# Properties

## Description

Gets or sets the human-readable description of the error. This is generated from the associated [MbError](#) using its `GetErrorMessage` method.

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

### Property Value

[string](#) ↗

## MbError

Gets or sets the [MbError](#) associated with this exception. This holds structured error information relevant to the exception.

```
public MbError MbError { get; set; }
```

### Property Value

[MbError](#)

# Namespace MB.Common.Jsons

## Classes

### [GenericJsonFileService](#)

Provides generic methods for saving and loading JSON files from disk.

# Class GenericJsonFileService

Namespace: [MB.Common.Jsons](#)

Assembly: MB.Common.dll

Provides generic methods for saving and loading JSON files from disk.

```
public static class GenericJsonFileService
```

## Inheritance

[object](#) ← GenericJsonFileService

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### EnsureFileExists(string)

Ensures that a file exists at the specified path, creating it if necessary.

```
public static MbResult<bool> EnsureFileExists(string filePath)
```

#### Parameters

**filePath** [string](#)

The full path of the file to check or create. Must not be null, empty, or whitespace.

#### Returns

[MbResult<bool>](#)

An [MbResult<T>](#) containing a boolean value indicating success or failure. Returns [true](#) if the file exists or was successfully created; otherwise, returns [false](#) with an error message.

#### Remarks

If the specified file does not exist, this method creates it along with any necessary directories. The created file will be initialized with an empty JSON array.

## Load<T>(string)

Loads JSON data from a file and deserializes it into an object of type `T`.

```
public static MbResult<T> Load<T>(string filePath) where T : new()
```

### Parameters

`filePath` [string](#)

The full path to the JSON file.

### Returns

[MbResult](#)<T>

An [MbResult](#)<T> containing the deserialized object, or an empty instance on failure.

### Type Parameters

`T`

The type to deserialize the JSON content into. Must have a parameterless constructor.

## Save<T>(T, string)

Saves the given data as JSON to the specified file path.

```
public static MbResult<bool> Save<T>(T data, string filePath)
```

### Parameters

`data` `T`

The data object to serialize and save.

`filePath` [string](#)

The full path where the file will be saved.

Returns

[MbResult<bool>](#)

An [MbResult<T>](#) indicating success or failure.

Type Parameters

T

The type of the data to serialize.

Remarks

Will automatically create the target directory if it doesn't exist.

# Namespace MB.Common.Parameters

## Classes

### [RevitParameterManager](#)

Manages shared parameters in a Revit document, ensuring required parameters exist and are properly configured.

# Class RevitParameterManager

Namespace: [MB.Common.Parameters](#)

Assembly: MB.Common.dll

Manages shared parameters in a Revit document, ensuring required parameters exist and are properly configured.

```
public class RevitParameterManager
```

## Inheritance

[object](#) ← RevitParameterManager

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### RevitParameterManager()

```
public RevitParameterManager()
```

## Methods

### EnsureParameters(Document, ToolsSharedParameter[])

Ensures that a set of shared parameters exists in the given Revit document. If any of the parameters are missing, they will be added during the transaction.

```
public static void EnsureParameters(Document doc, ToolsSharedParameter[] parameters)
```

## Parameters

**doc** Document

The Revit Autodesk.Revit.DB.Document in which the parameters should be ensured.

**parameters** [ToolsSharedParameter\[\]](#)

An array of [ToolsSharedParameter](#) representing the shared parameters to verify or add.

## EnsureParameters(Document, Mbftool)

Ensures that all required shared parameters for a specific tool exist in the Revit document.

```
public static void EnsureParameters(Document doc, Mbftool tool)
```

### Parameters

**doc** Document

The Revit document.

**tool** [Mbftool](#)

The tool requiring the shared parameters.

# Namespace MB.Common.Services

## Classes

### [EmbeddedResourceService](#)

Provides functionality for accessing and processing embedded resources in assemblies.

### [MailService](#)

Provides methods for sending emails, including support for attachments and both synchronous and asynchronous operations.

# Class EmbeddedResourceService

Namespace: [MB.Common.Services](#)

Assembly: MB.Common.dll

Provides functionality for accessing and processing embedded resources in assemblies.

```
public static class EmbeddedResourceService
```

## Inheritance

[object](#) ← EmbeddedResourceService

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### LoadEmbeddedResourceAsync<T>(string, Assembly)

Loads and deserializes an embedded JSON resource file into a specified type.

```
public static Task<T> LoadEmbeddedResourceAsync<T>(string resourceName, Assembly assembly)
```

#### Parameters

resourceName [string](#)

The fully qualified name of the embedded resource

assembly [Assembly](#)

#### Returns

[Task](#)<T>

Deserialized object of type T from the JSON resource

#### Type Parameters

T

The type to deserialize the JSON content into

Exceptions

FileNotFoundException

Thrown when the specified embedded resource is not found

# Class MailService

Namespace: [MB.Common.Services](#)

Assembly: MB.Common.dll

Provides methods for sending emails, including support for attachments and both synchronous and asynchronous operations.

```
public static class MailService
```

## Inheritance

[object](#) ← MailService

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### Send(string, List<string>)

Sends an email synchronously with optional attachments. Uses a predefined sender and receiver email configuration.

```
[LogAspect]  
public static void Send(string msg, List<string> attachmentPaths = null)
```

#### Parameters

**msg** [string](#)

The body of the email message.

**attachmentPaths** [List](#)<[string](#)>

Optional list of file paths to attach to the email.

### SendAsync(string, List<string>)

Sends an email asynchronously with optional attachments. Useful for avoiding UI blocking during email transmission.

```
[LogAspect]  
public static Task SendAsync(string msg, List<string> attachmentPaths = null)
```

## Parameters

**msg** [string](#)

The body of the email message.

**attachmentPaths** [List](#)<[string](#)>

Optional list of file paths to attach to the email.

## Returns

[Task](#)

# Namespace MB.Common.Settings

## Classes

### [ParameterData](#)

Represents a parameter with a name, display name, and selection state.

# Class ParameterData

Namespace: [MB.Common.Settings](#)

Assembly: MB.Common.dll

Represents a parameter with a name, display name, and selection state.

```
public class ParameterData : ObservableObject, INotifyPropertyChanged,  
INotifyPropertyChanging
```

## Inheritance

[object](#) ← [ObservableObject](#) ← ParameterData

## Implements

[INotifyPropertyChanged](#), [INotifyPropertyChanging](#)

## Inherited Members

[ObservableObject.OnPropertyChanged\(PropertyChangedEventArgs\)](#) ,  
[ObservableObject.OnPropertyChanging\(PropertyChangingEventArgs\)](#) ,  
[ObservableObject.OnPropertyChanged\(string\)](#) , [ObservableObject.OnPropertyChanging\(string\)](#) ,  
[ObservableObject SetProperty<T>\(ref T, T, string\)](#) ,  
[ObservableObject SetProperty<T>\(ref T, T, IEqualityComparer<T>, string\)](#) ,  
[ObservableObject SetProperty<T>\(T, T, Action<T>, string\)](#) ,  
[ObservableObject SetProperty<T>\(T, T, IEqualityComparer<T>, Action<T>, string\)](#) ,  
[ObservableObject SetProperty<TModel, T>\(T, T, TModel, Action<TModel, T>, string\)](#) ,  
[ObservableObject SetProperty<TModel, T>\(T, T, IEqualityComparer<T>, TModel, Action<TModel, T>, string\)](#) ,  
[ObservableObject SetPropertyAndNotifyOnCompletion\(ref ObservableObject.TaskNotifier, Task, string\)](#) ,  
  
[ObservableObject SetPropertyAndNotifyOnCompletion\(ref ObservableObject.TaskNotifier, Task, Action<Task>, string\)](#) ,  
[ObservableObject SetPropertyAndNotifyOnCompletion<T>\(ref ObservableObject.TaskNotifier<T>, Task<T>, string\)](#) ,  
[ObservableObject SetPropertyAndNotifyOnCompletion<T>\(ref ObservableObject.TaskNotifier<T>, Task<T>, Action<Task<T>>, string\)](#) ,  
[ObservableObject.PropertyChanged](#) , [ObservableObject.PropertyChanging](#) , [object.ToString\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

# Constructors

## ParameterData(string, string, bool)

Represents a parameter with a name, display name, and selection state.

```
public ParameterData(string name, string displayName, bool isSelected = false)
```

### Parameters

**name** [string](#)

The internal name of the parameter.

**displayName** [string](#)

The display name of the parameter.

**isSelected** [bool](#)

Indicates whether the parameter is selected.

# Properties

## DisplayName

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

### Property Value

[string](#)

## IsSelected

```
public bool IsSelected { get; set; }
```

### Property Value

[bool](#)

## Name

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

## Property Value

[string](#) ↗

# Namespace MB.Common.Settings.Listing Settings

## Classes

### [FittingGroupHelper](#)

Provides helper methods for grouping [StandardFittingType](#) values into their corresponding PDF group names for listing and reporting purposes.

### [ListingConfig](#)

Represents the configuration for a listing, including elements, family details, category, and fitting types. This class provides constructors to initialize its properties either from a collection of elements or from a Data Transfer Object (DTO).

### [ListingDto](#)

Represents a data transfer object for a listing, including details such as family name, symbol name, category, and selected fitting type.

### [ListingMapper](#)

Provides methods for mapping between [ListingConfig](#) and [ListingDto](#) objects.

### [ListingSettingsHelper](#)

Provides helper methods for working with listing settings, including retrieval of [ListingDto](#) objects based on family and type information.

### [MicroBimListingRecommendationDto](#)

Represents a Data Transfer Object (DTO) used for providing recommendations specific to Micro BIM listings.

### [StandardFittingGroups](#)

Provides predefined groupings of standard fitting types for different fitting categories.

### [ValueToStandardTypeDto](#)

Represents a Data Transfer Object (DTO) that associates a specific value with a standard fitting type.

## Enums

### [StandardFittingType](#)

Represents a collection of fitting types commonly utilized within piping systems, categorized by different connection methods such as threaded, grooved, embedded couplings, integrated systems (IGS), flanged connections, and more.

# Class FittingGroupHelper

Namespace: [MB.Common.Settings.ListingSettings](#)

Assembly: MB.Common.dll

Provides helper methods for grouping [StandardFittingType](#) values into their corresponding PDF group names for listing and reporting purposes.

```
public static class FittingGroupHelper
```

## Inheritance

[object](#) ← FittingGroupHelper

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## GetPdfGroupName(StandardFittingType)

Gets the PDF group name for a specified [StandardFittingType](#).

```
public static string GetPdfGroupName(StandardFittingType type)
```

### Parameters

**type** [StandardFittingType](#)

The fitting type to evaluate.

### Returns

[string](#)

The group name as a [string](#) if the type is recognized; otherwise, [null](#). Possible group names include "Threaded Fittings", "Grooved Fittings", "Welded Fittings", "IGS Fittings", "Flanged Fittings", and "Loose Outlets".

# Class ListingConfig

Namespace: [MB.Common.Settings.ListingSettings](#)

Assembly: MB.Common.dll

Represents the configuration for a listing, including elements, family details, category, and fitting types. This class provides constructors to initialize its properties either from a collection of elements or from a Data Transfer Object (DTO).

```
public class ListingConfig : ObservableObject, INotifyPropertyChanged,  
INotifyPropertyChanging
```

## Inheritance

[object](#) ← [ObservableObject](#) ← ListingConfig

## Implements

[INotifyPropertyChanged](#), [INotifyPropertyChanging](#)

## Inherited Members

[ObservableObject.OnPropertyChanged\(PropertyChangedEventArgs\)](#) ,  
[ObservableObject.OnPropertyChanging\(PropertyChangingEventArgs\)](#) ,  
[ObservableObject.OnPropertyChanged\(string\)](#) , [ObservableObject.OnPropertyChanging\(string\)](#) ,  
[ObservableObject SetProperty<T>\(ref T, T, string\)](#) ,  
[ObservableObject SetProperty<T>\(ref T, T, IEqualityComparer<T>, string\)](#) ,  
[ObservableObject SetProperty<T>\(T, T, Action<T>, string\)](#) ,  
[ObservableObject SetProperty<T>\(T, T, IEqualityComparer<T>, Action<T>, string\)](#) ,  
[ObservableObject SetProperty<TModel, T>\(T, T, TModel, Action<TModel, T>, string\)](#) ,  
[ObservableObject SetProperty<TModel, T>\(T, T, IEqualityComparer<T>, TModel, Action<TModel, T>, string\)](#) ,  
[ObservableObject SetPropertyAndNotifyOnCompletion\(ref ObservableObject.TaskNotifier, Task, string\)](#) ,  
  
[ObservableObject SetPropertyAndNotifyOnCompletion\(ref ObservableObject.TaskNotifier, Task, Action<Task>, string\)](#) ,  
[ObservableObject SetPropertyAndNotifyOnCompletion<T>\(ref ObservableObject.TaskNotifier<T>, Task<T>, string\)](#) ,  
[ObservableObject SetPropertyAndNotifyOnCompletion<T>\(ref ObservableObject.TaskNotifier<T>, Task<T>, Action<Task<T>>, string\)](#) ,  
[ObservableObject.PropertyChanged](#) , [ObservableObject.PropertyChanging](#) , [object.ToString\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

# Constructors

## ListingConfig(ListingDto)

Represents a configuration for a listing, managing elements, family names, symbol names, and associated metadata.

```
public ListingConfig(ListingDto dto)
```

Parameters

`dto` [ListingDto](#)

## ListingConfig(List<FamilyInstance>, StandardFittingType)

Represents the configuration of a listing, including information about the elements, fitting type, category, and suggested values.

```
public ListingConfig(List<FamilyInstance> elements, StandardFittingType standardFitting  
= StandardFittingType.None)
```

Parameters

`elements` [List](#)<FamilyInstance>

`standardFitting` [StandardFittingType](#)

Remarks

This class sets the initial configuration for a list of elements including family and symbol names, category, fitting type, and other properties based on a representative element from the provided list.

## ListingConfig(string, StandardFittingType)

Represents the configuration for a listing, including the handling of family and symbol names, associated category, selected fitting type, and suggested values, derived from the specified inputs.

```
public ListingConfig(string name, StandardFittingType selectedFittingType)
```

Parameters

`name` [string](#)

`selectedFittingType` [StandardFittingType](#)

## Remarks

This class processes the provided name and fitting type to initialize properties such as family name, symbol name, full name, and the category associated with the represented elements. It also manages suggested values and determines whether a valid value is selected.

# Properties

## CategoryName

`public string CategoryName { get; set; }`

## Property Value

[string](#)

## Count

`public int Count { get; set; }`

## Property Value

[int](#)

## Elements

Gets the collection of FamilyInstance objects associated with the current listing configuration.

`public List<FamilyInstance> Elements { get; }`

## Property Value

[List](#)<[FamilyInstance](#)>

## Remarks

This property holds an immutable list of family instance elements, typically used to represent a grouped set of related elements in a listing configuration. The collection is initialized during the creation of the [ListingConfig](#) and cannot be modified outside the class.

## FamilyName

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

### Property Value

[string](#) ↗

## HasValue

```
public bool HasValue { get; set; }
```

### Property Value

[bool](#) ↗

## Name

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

### Property Value

[string](#) ↗

## SelectedFittingType

```
public StandardFittingType SelectedFittingType { get; set; }
```

Property Value

[StandardFittingType](#)

## SuggestedValues

```
public List<StandardFittingType> SuggestedValues { get; set; }
```

Property Value

[List](#) <[StandardFittingType](#)>

## SymbolName

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

Property Value

[string](#)

# Class ListingDto

Namespace: [MB.Common.Settings.ListingSettings](#)

Assembly: MB.Common.dll

Represents a data transfer object for a listing, including details such as family name, symbol name, category, and selected fitting type.

```
public class ListingDto
```

## Inheritance

[object](#) ← ListingDto

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### ListingDto(string, string, string, string, StandardFittingType)

Initializes a new instance of the [ListingDto](#) class.

```
public ListingDto(string familyName, string symbolName, string name, string categoryName,  
StandardFittingType selectedFittingType)
```

## Parameters

### familyName [string](#)

The name of the Revit family.

### symbolName [string](#)

The name of the family type or symbol.

### name [string](#)

The display name or identifier for the listing.

`categoryName` [string](#)

The name of the Revit category.

`selectedFittingType` [StandardFittingType](#)

The selected standard fitting type.

## Properties

### CategoryName

Gets or sets the name of the Revit category.

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

### Property Value

[string](#)

### FamilyName

Gets or sets the name of the family in Revit.

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

### Property Value

[string](#)

### Name

Gets or sets the display name or identifier for the listing.

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

### Property Value

[string](#)

## SelectedFittingType

Gets or sets the selected standard fitting type for the element.

```
public StandardFittingType SelectedFittingType { get; set; }
```

Property Value

[StandardFittingType](#)

## SymbolName

Gets or sets the name of the family type or symbol.

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

Property Value

[string](#) ↗

# Class ListingMapper

Namespace: [MB.Common.Settings.ListingSettings](#)

Assembly: MB.Common.dll

Provides methods for mapping between [ListingConfig](#) and [ListingDto](#) objects.

```
public class ListingMapper
```

## Inheritance

[object](#) ← ListingMapper

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class contains static methods to facilitate the conversion of data between the [ListingConfig](#) and [ListingDto](#) types. These methods are useful for scenarios where data needs to be transferred between different layers of an application, such as mapping configuration data to a data transfer object (DTO) for use in APIs or other external systems.

## Constructors

### ListingMapper()

```
public ListingMapper()
```

## Methods

### FromDto(ListingDto)

Converts a [ListingDto](#) object into a [ListingConfig](#) object.

```
public static ListingConfig FromDto(ListingDto dto)
```

## Parameters

**dto** [ListingDto](#)

The source [ListingDto](#) instance to be converted.

## Returns

[ListingConfig](#)

A new instance of [ListingConfig](#) containing the mapped data from the provided [ListingDto](#).

## ToDto(ListingConfig)

Converts a [ListingConfig](#) object into a [ListingDto](#) object.

```
public static ListingDto ToDto(ListingConfig config)
```

## Parameters

**config** [ListingConfig](#)

The source [ListingConfig](#) instance to be converted.

## Returns

[ListingDto](#)

A new instance of [ListingDto](#) containing the mapped data from the provided [ListingConfig](#).

# Class ListingSettingsHelper

Namespace: [MB.Common.Settings.ListingSettings](#)

Assembly: MB.Common.dll

Provides helper methods for working with listing settings, including retrieval of [ListingDto](#) objects based on family and type information.

```
public static class ListingSettingsHelper
```

## Inheritance

[object](#) ← ListingSettingsHelper

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## GetSelectedListingDto(string)

Retrieves the [ListingDto](#) that matches the specified family name and type from the user listing settings.

```
public static ListingDto? GetSelectedListingDto(string familyNameAndType)
```

### Parameters

**familyNameAndType** [string](#)

The combined family name and type to search for.

### Returns

[ListingDto](#)

The [ListingDto](#) that matches the specified **familyNameAndType**, or **null** if no match is found or if the user listing settings could not be loaded.

# Class MicroBimListingRecommendationDto

Namespace: [MB.Common.Settings.ListingSettings](#)

Assembly: MB.Common.dll

Represents a Data Transfer Object (DTO) used for providing recommendations specific to Micro BIM listings.

```
public class MicroBimListingRecommendationDto
```

## Inheritance

[object](#) ← MicroBimListingRecommendationDto

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### MicroBimListingRecommendationDto()

```
public MicroBimListingRecommendationDto()
```

## Properties

### Name

Gets or sets the name, which must follow the format "FamilyName-SymbolName".

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

### Property Value

[string](#)

## SuggestedType

Represents the suggested fitting type for a listing recommendation. This property must follow the format "FamilyName-SymbolName".

```
public StandardFittingType SuggestedType { get; set; }
```

### Property Value

[StandardFittingType](#)

# Class StandardFittingGroups

Namespace: [MB.Common.Settings.ListingSettings](#)

Assembly: MB.Common.dll

Provides predefined groupings of standard fitting types for different fitting categories.

```
public static class StandardFittingGroups
```

## Inheritance

[object](#) ← StandardFittingGroups

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Fields

### ElbowFittings

Gets a read-only collection of elbow-type fittings, including threaded, grooved, and embedded coupling variants.

```
public static readonly IReadOnlyList<StandardFittingType> ElbowFittings
```

#### Field Value

[IReadOnlyList](#)<[StandardFittingType](#)>

### RemainingFittings

Gets a read-only collection of miscellaneous fittings, including couplings, caps, reducers, and general-purpose fittings.

```
public static readonly IReadOnlyList<StandardFittingType> RemainingFittings
```

#### Field Value

[IReadOnlyList](#) <[StandardFittingType](#)>

## SpudFittings

Gets a read-only collection of spud fittings, including mechanical tees, outlets, and branch connections.

```
public static readonly IReadOnlyList<StandardFittingType> SpudFittings
```

### Field Value

[IReadOnlyList](#) <[StandardFittingType](#)>

## TeeFittings

Gets a read-only collection of tee and cross fittings, including threaded, grooved, and embedded coupling variants.

```
public static readonly IReadOnlyList<StandardFittingType> TeeFittings
```

### Field Value

[IReadOnlyList](#) <[StandardFittingType](#)>

# Enum StandardFittingType

Namespace: [MB.Common.Settings.ListingSettings](#)

Assembly: MB.Common.dll

Represents a collection of fitting types commonly utilized within piping systems, categorized by different connection methods such as threaded, grooved, embedded couplings, integrated systems (IGS), flanged connections, and more.

```
[TypeConverter(typeof(EnumDescriptionTypeConverter))]  
public enum StandardFittingType
```

## Fields

**BlankEnd = 38**

Represents a fitting type that is identified as a blank end.

**Bush = 10**

Represents the "Bush" fitting type in the StandardFittingType enumeration.

**CrossEmbeddedCoupling = 34**

Represents a cross fitting type that includes an embedded coupling. This fitting type is typically used in plumbing or piping systems where a cross layout, integrated with an embedded coupling, is required to connect multiple pipes or components.

**ElbowWithEmbeddedCoupling = 3**

Represents a type of fitting defined as "Elbow with Embedded Coupling." This fitting combines an elbow and a coupling into a single embedded component, providing a seamless transition and integrated functionality within piping systems.

**FlangedFittingGeneral = 57**

Represents a general type of flanged fitting in the enumeration of standard fitting types. This member is used to categorize fittings that utilize a flange connection, typically offering a secure and durable method of joining pipes or other components.

**FullSocket = 12**

Represents a full socket fitting type in the [StandardFittingType](#) enumeration. A full socket is typically used to provide a secure and complete connection in piping systems, allowing joining of components with an extended and enclosed fitting interface.

**GRVFlangedFittingGeneral = 59**

Represents a grooved flanged fitting general type in the [StandardFittingType](#) enumeration.

**GRVTRDFittingGeneral = 49**

Represents a grooved fitting of the TRD (Threaded Rotational Design) type within the StandardFittingType enumeration. This type is used to categorize general-purpose grooved TRD fittings.

**GroovedBranchOletGOLBO = 6**

Represents a grooved branch olet fitting, identified with the abbreviation GOL BO. This type of fitting is commonly used in piping systems to provide a branch connection with grooved ends for easy installation and alignment.

**GroovedCap = 41**

Represents a grooved cap type fitting in the context of standard fitting types. Used for applications requiring a grooved cap connection.

**GroovedCoupling = 14**

Represents the Grooved Coupling fitting type in the StandardFittingType enumeration. The Grooved Coupling is a standard fitting type used in piping systems. It utilizes a grooved design to enable secure and efficient assembly of piping connections, ensuring structural integrity and ease of installation.

**GroovedCross = 9**

Represents a grooved cross fitting type commonly used in piping systems. This fitting is designed with grooves that enable secure coupling and assembly within grooved pipe systems. It typically serves as a four-way connection point for pipes in a cross configuration, ensuring efficient fluid distribution.

**GroovedElbow = 44**

Represents a grooved elbow fitting type within the StandardFittingType enumeration. A grooved elbow is typically used in piping systems to change the direction of flow while maintaining a grooved connection for easy assembly and disassembly of components.

**GroovedFittingGeneral = 46**

Represents a general fitting type for grooved connections. This enum member is used to identify standard fittings that align with the classification of grooved fittings without specifying a more precise category.

#### **GroovedMechanicalCross = 18**

Represents a grooved mechanical cross fitting type, typically used in piping systems. This fitting allows four pipes to be joined at right angles to one another, facilitating mechanical joint installations with grooved connections.

#### **GroovedMechanicalTee = 16**

Represents a grooved mechanical tee fitting type, used in piping systems.

#### **GroovedReducer = 42**

Represents a grooved reducer fitting type within the [StandardFittingType](#) enumeration. A grooved reducer is a type of pipe fitting used to connect two pipes of different diameters while employing a grooved connection mechanism.

#### **GroovedTee = 45**

Represents a grooved tee fitting type. This fitting is characterized by its grooved ends and branch connections, commonly used for mechanical pipe joining systems.

#### **HalfSocket = 11**

Represents a standard fitting type "Half Socket". This enum value is used to represent a half socket fitting in various configurations within the system. It is identified by the enum value of 11 and has a display name of "Half Socket".

#### **HexNipple = 13**

Represents a hex nipple fitting type in the [StandardFittingType](#) enumeration. A hex nipple is a threaded fitting with hexagonal sides, designed for connecting two pipes or fittings. It is commonly used in piping systems to create secure and leak-proof connections.

#### **IGSCap = 24**

Represents an IGS-specific cap fitting type in the standard fitting type enumeration. This fitting is used to seal the end of a pipe or other component in a system that uses IGS (Integrated Grooved System) fittings.

#### **IGSCoupling = 22**

Represents the "IGS Coupling" fitting type in the StandardFittingType enumeration. This fitting type is part of the Integrated Grooved System (IGS), designed to provide secure connections in piping systems using the coupling mechanism.

#### **IGSElbowWithEmbeddedCoupling = 20**

Represents an IGS elbow fitting that includes an embedded coupling. This type of fitting is commonly used in piping systems to connect two pipe segments at an angle while providing an integrated coupling for secure and efficient connections.

#### **IGSEndOlet = 58**

Represents a fitting type categorized as "IGS End Olet". This type is designed as part of the IGS (Integrated Grooved System) fittings and typically serves as an end connection or branch outlet in piping systems.

#### **IGSFittingGeneral = 60**

Represents a general fitting type for the IGS (International Groove System) standard. This designation is used when a specific IGS fitting does not have a more detailed classification within the enumeration.

#### **IGSGRVFittingGeneral = 54**

Represents the "IGS GRV Fitting General" type within the [StandardFittingType](#) enumeration. It is used to denote a general category of grooved fittings within the IGS system.

#### **IGSIRFittingGeneral = 48**

Represents a general fitting type for the IGS and IR classifications. This enumeration member is part of the StandardFittingType enum, used for identifying fittings that combine characteristics or applications of both IGS (Integrated Groove System) and IR (Internal Resistance).

#### **IGSMechanicalTee = 25**

Represents an IGS Mechanical Tee fitting type within the [StandardFittingType](#) enumeration. An IGS Mechanical Tee is a type of fitting used in piping systems that typically allows for the branching of a pipeline while maintaining a mechanical connection.

#### **IGSOlet = 23**

Represents the IGS Olet fitting type in the [StandardFittingType](#) enumeration. This type is used for fittings designed to connect or branch pipes within the IGS (Integrated Grooved System) category.

#### **IGSReducer = 27**

Represents the IGS Reducer fitting type in the [StandardFittingType](#) enumeration. This fitting type is specific to IGS (Integrated Grooved System) and is used when a reducer is required to transition between pipes of different diameters in a grooved piping system.

#### **IGSTRDFittingGeneral = 52**

Represents the general fitting type for IGS threaded connections. This enum member is used to categorize fittings with threaded specifications under the IGS (Integrated Grooved System) standards.

#### **IGSTeeEmbeddedCoupling = 21**

Represents a fitting type categorized as IGS Tee with Embedded Coupling. This fitting type is typically used in piping or plumbing systems where an IGS-standard tee fitting incorporates an embedded coupling for streamlined and integrated connections.

#### **IRCap = 33**

Represents a standard fitting type classified as "IR Cap".

#### **IRCoupling = 29**

Represents the IR Coupling fitting type used in piping configurations. Corresponds to the fitting type with the designation "IR Coupling". Identified by its unique enum value within the StandardFittingType enumeration.

#### **IРЕlbowEmbeddedCoupling = 30**

Represents an IR (Integrated Rolled) elbow fitting with an embedded coupling. This type of fitting is designed to connect two pipes at an angle, while integrating a coupling within the fitting itself to facilitate secure and efficient connections.

#### **IRGroovedEndOlet = 28**

Represents an IR Grooved End Olet fitting type. This fitting is typically used in piping systems where grooved end connections are required to facilitate assembly without welding.

#### **IRRReducer = 32**

Represents a specific type of fitting known as "IR Reducer," which is part of the "IR" (Integral Reducer) grouping within the available standard fitting types. This type is typically used to connect pipes of different diameters while adhering to the specifications of the "IR" fitting standard.

#### **IRTeeEmbeddedCoupling = 31**

Represents a type of fitting identified as "IR Tee with Embedded Coupling". This fitting type is typically used for connection systems where a Tee shape with an integrated coupling mechanism is required for industrial or plumbing applications under the IR (InfraRed) fitting category.

**MechTee922TRD = 55**

Represents the "Mechanical Tee 922 TRD" fitting type in the [StandardFittingType](#) enumeration. This fitting type typically refers to a specific threaded mechanical tee designed for use in piping systems.

**ModelingHelperNonConnector = 43**

Represents a placeholder or helper fitting type that is not connected to any actual model or component. It can be used as a utility within modeling processes where a non-functional connector type is needed.

**NewMapping = 36**

Represents a new mapping type of fitting in the [StandardFittingType](#) enum.

**None = 50**

Represents an unspecified or default fitting type in the [StandardFittingType](#) enumeration. This value is used when no specific fitting type is applicable or has not been determined.

**PVCfittingGeneralPlainEnd = 56**

Represents a general-purpose PVC fitting with a plain end.

**PipeOletTap = 5**

Represents the "Pipe Olet Tap" type in the [StandardFittingType](#) enumeration. This fitting is typically used for branching off the main pipe, allowing for the connection of smaller pipes or outlets.

**Plug = 37**

Represents a plug fitting type. This fitting is typically used to seal or close the end of a pipe or fitting, preventing fluid or gas flow.

**PlugConnectedAfterFullSocket = 35**

Represents a standard fitting type where a plug is connected after the application of a full socket.

**PlugConnectedAfterHalfSocket = 7**

Represents a plug that is connected after a half socket fitting. This type is typically used in pipeline or plumbing systems to seal or terminate a connection while attached to a half socket configuration.

## **TeeWithEmbeddedCoupling = 4**

Represents a tee fitting that includes an embedded coupling. This fitting type combines the functionality of a tee and a coupling, offering a streamlined design for specific pipe connection scenarios.

## **ThreadedCap = 51**

Represents a threaded cap fitting type.

## **ThreadedCouplingUS = 15**

Represents a threaded coupling fitting specific to U.S. standards.

## **ThreadedCross = 8**

Represents a fitting type classified as "Threaded Cross". This fitting type is designed with threaded connections, allowing for a cross-shaped configuration that provides four-way pipe connection points.

## **ThreadedElbow = 1**

Represents a threaded elbow fitting type within the standard fitting category. Threaded elbow fittings are designed to join two pipes at an angle, typically 90 or 45 degrees, and utilize threading for connection.

## **ThreadedFittingGeneral = 47**

Represents a general threaded fitting type. It is a category within the [StandardFittingType](#) enumeration that serves as a generic descriptor for threaded fitting components where no specific subtype is identified or required.

## **ThreadedMechanicalCross = 19**

Specifies a fitting type that represents a threaded mechanical cross.

## **ThreadedMechanicalTee = 17**

Represents a threaded mechanical tee fitting type within the [StandardFittingType](#) enumeration. The ThreadedMechanicalTee is a type of pipe fitting used to connect branches of piping systems. It features threaded connections for secure assembly in mechanical piping configurations.

## **ThreadedTee = 2**

Represents a type of standard fitting known as "Threaded Tee." This fitting features a T-shaped design with threaded connections at all three connection points, commonly used in piping systems for branching a flow into two lines or combining two flows into one.

**WGRVFittingGeneral = 53**

Represents a welded grooved fitting general type in the [StandardFittingType](#) enumeration.

**WeldedEndFittingGeneral = 40**

Represents a generalized welded end fitting type. This enumeration member is used to categorize fittings that utilize welded connections for establishing a secure link between piping systems and components.

**WeldedGroovedPieceIR = 39**

Represents a welded fitting type with grooved connections for the IR (In-Room) standard. This enum member is part of the [StandardFittingType](#) enumeration and is commonly used for creating or identifying welded fittings with grooved design utilized in IR-specific applications or configurations in mechanical systems.

# Class ValueToStandardTypeDto

Namespace: [MB.Common.Settings.ListingSettings](#)

Assembly: MB.Common.dll

Represents a Data Transfer Object (DTO) that associates a specific value with a standard fitting type.

```
public class ValueToStandardTypeDto
```

## Inheritance

[object](#) ← ValueToStandardTypeDto

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### ValueToStandardTypeDto()

```
public ValueToStandardTypeDto()
```

## Properties

### SuggestedType

Gets or sets the standard type associated with the value to standard type mapping.

```
public StandardFittingType SuggestedType { get; set; }
```

### Property Value

[StandardFittingType](#)

## Value

Gets or sets the value associated with the mapping to a standard fitting type.

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

Property Value

[string](#) ↗

# Namespace MB.Common.SharedParameters

## Classes

### [ElementExtensions](#)

Provides extension methods for working with Revit Element objects.

### [SharedParameterExtensions](#)

Provides extension methods for the [ToolsSharedParameter](#) class.

## Enums

### [SharedParameterType](#)

Represents the types of shared parameters that can be used in Revit.

### [ToolsSharedParameter](#)

Defines shared parameters for use within Revit, including metadata such as GUIDs, descriptions, parameter types, and applicable categories.

# Class ElementExtensions

Namespace: [MB.Common.SharedParameters](#)

Assembly: MB.Common.dll

Provides extension methods for working with Revit Element objects.

```
public static class ElementExtensions
```

## Inheritance

[object](#) ← ElementExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## MbGetParameter(Element, ToolsSharedParameter)

Retrieves a Revit Autodesk.Revit.DB.Parameter associated with the specified shared parameter from the given Element.

```
public static Parameter MbGetParameter(this Element element, ToolsSharedParameter  
sharedParameter)
```

### Parameters

**element** Element

The Revit Element to retrieve the parameter from.

**sharedParameter** [ToolsSharedParameter](#)

The [ToolsSharedParameter](#) to retrieve.

### Returns

Parameter

The Autodesk.Revit.DB.Parameter object associated with the specified shared parameter. Returns `null` if the parameter is not found.

## MbSetParameter<T>(Element, ToolsSharedParameter, T)

Sets the value of a Revit Autodesk.Revit.DB.Parameter associated with a specified shared parameter.

```
public static void MbSetParameter<T>(this Element element, ToolsSharedParameter  
sharedParameter, T value)
```

### Parameters

**element** Element

The Revit Element to set the parameter for.

**sharedParameter** [ToolsSharedParameter](#)

The [ToolsSharedParameter](#) whose value is to be set.

**value** T

The value to assign to the parameter.

### Type Parameters

T

The type of the value to set (e.g., int, double, string, ElementId).

### Exceptions

**ArgumentException**

Thrown if the parameter value type is unsupported or the parameter cannot be set with the specified type.

# Class SharedParameterExtensions

Namespace: [MB.Common.SharedParameters](#)

Assembly: MB.Common.dll

Provides extension methods for the [ToolsSharedParameter](#) class.

```
public static class SharedParameterExtensions
```

## Inheritance

[object](#) ← SharedParameterExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### GetCreationOptions(ToolsSharedParameter)

Creates an instance of ExternalDefinitionCreationOptions for a [ToolsSharedParameter](#).

```
public static ExternalDefinitionCreationOptions GetCreationOptions(this  
ToolsSharedParameter sharedParameter)
```

#### Parameters

sharedParameter [ToolsSharedParameter](#)

The shared parameter.

#### Returns

ExternalDefinitionCreationOptions

An ExternalDefinitionCreationOptions instance.

## GetDisplayName(ToolsSharedParameter)

Retrieves the display name of a [ToolsSharedParameter](#).

```
public static string GetDisplayName(this ToolsSharedParameter sharedParameter)
```

### Parameters

**sharedParameter** [ToolsSharedParameter](#)

The shared parameter.

### Returns

[string](#)

The display name of the shared parameter.

## GetGuid(ToolsSharedParameter)

Retrieves the GUID associated with a [ToolsSharedParameter](#).

```
public static string GetGuid(this ToolsSharedParameter sharedParameter)
```

### Parameters

**sharedParameter** [ToolsSharedParameter](#)

The shared parameter.

### Returns

[string](#)

The GUID of the shared parameter.

## GetIncludedCategories(ToolsSharedParameter)

Retrieves the list of categories included in a [ToolsSharedParameter](#).

```
public static List<BuiltInCategory> GetIncludedCategories(this  
ToolsSharedParameter sharedParameter)
```

## Parameters

**sharedParameter** [ToolsSharedParameter](#)

The shared parameter.

## Returns

[List](#) <BuiltInCategory>

A list of BuiltInCategory values.

## GetSharedParameterType(ToolsSharedParameter)

Retrieves the type of [ToolsSharedParameter](#).

```
public static SharedParameterType GetSharedParameterType(this ToolsSharedParameter  
sharedParameter)
```

## Parameters

**sharedParameter** [ToolsSharedParameter](#)

The shared parameter.

## Returns

[SharedParameterType](#)

The [SharedParameterType](#) of the shared parameter.

# Enum SharedParameterType

Namespace: [MB.Common.SharedParameters](#)

Assembly: MB.Common.dll

Represents the types of shared parameters that can be used in Revit.

```
public enum SharedParameterType
```

## Fields

**Area = 3**

Represents an area parameter type, used for values that define area measurements.

**Integer = 2**

Represents an integer parameter type, used for whole number values.

**Text = 1**

Represents a text parameter type, used for string values.

**YesOrNo = 0**

Represents a Yes/No parameter type, typically used for boolean values.

# Enum ToolsSharedParameter

Namespace: [MB.Common.SharedParameters](#)

Assembly: MB.Common.dll

Defines shared parameters for use within Revit, including metadata such as GUIDs, descriptions, parameter types, and applicable categories.

```
public enum ToolsSharedParameter
```

## Extension Methods

[SharedParameterExtensions.GetCreationOptions\(ToolsSharedParameter\)](#) ,  
[SharedParameterExtensions.GetDisplayName\(ToolsSharedParameter\)](#) ,  
[SharedParameterExtensions.GetGuid\(ToolsSharedParameter\)](#) ,  
[SharedParameterExtensions.GetIncludedCategories\(ToolsSharedParameter\)](#) ,  
[SharedParameterExtensions.GetSharedParameterType\(ToolsSharedParameter\)](#).

## Fields

[Guid("9fdf6b2a-2772-4194-969c-f46d6e6ab866")]  
[SharedParameterType(SharedParameterType.Area)] [RevitCategories] ActualCoverage = 1

Represents the "MBF\_ActualCoverage" shared parameter. This shared parameter is applicable to pipe curves, pipe fittings, sprinklers, and pipe accessories. It is of type [Area](#).

[Guid("282d1a77-f651-4479-b6eb-85d84361c615")]  
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] AddFitToId = 9

Represents the "MBF\_AddFitToID" shared parameter. This shared parameter is applicable to pipe curves, pipe fittings, pipe accessories, generic models, model groups, mechanical equipment, sprinklers, fire alarm devices, and flex pipe curves. It is of type [Text](#).

[Guid("40221BB0-FFFE-42C6-BFF1-F1048AFA0664")]  
[SharedParameterType(SharedParameterType.Integer)] [RevitCategories] Allowance = 17

Represents the "MBF\_Allowance" shared parameter. This shared parameter is applicable to flex pipe curves. It is of type [Integer](#).

[Guid("6BEC8CDC-378E-4389-A7D6-EAF2493CEB9A")]  
[SharedParameterType(SharedParameterType.YesNo)] [RevitCategories] CalculateSum =

Represents the "MBF\_IsFitting" shared parameter. This shared parameter is used for flex pipe curves and is of type [YesOrNo](#).

```
[Guid("14fca12a-1224-41e5-a8a6-d4c0af178974")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Color = 7
```

Represents the "MBF\_Color" shared parameter. This shared parameter is applicable to pipe curves, pipe fittings, pipe accessories, generic models, model groups, mechanical equipment, sprinklers, fire alarm devices, and flex pipe curves. It is of type [Text](#).

```
[Guid("27e7c7ba-40cc-4ca6-b0c3-07a8e537a3bb")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories]
DescriptionAndAbbreviationListNo = 11
```

Represents the "MBF\_DescAbbrvListNo" shared parameter. This shared parameter is applicable to pipe curves, pipe fittings, pipe accessories, generic models, model groups, mechanical equipment, sprinklers, fire alarm devices, and flex pipe curves. It is of type [Text](#).

```
[Guid("2e46e681-4255-4016-9e53-722cb8aaaf32")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] DischargeDensity =
43
```

Represents the discharge density parameter used in Revit for various categories, such as pipes, fittings, and mechanical equipment. This shared parameter is identified by the GUID `2e46e681-4255-4016-9e53-722cb8aaaf32` and is categorized as text. It is applicable to multiple Revit categories, including pipe curves, pipe fittings, pipe accessories, generic models, flexible pipe curves, sprinklers, and mechanical equipment.

```
[Guid("ed2a2b7d-0425-4ed2-a2e1-85bb3c602c9e")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] DischargeDiameter =
47
```

Represents the K-factor parameter used in mechanical and fire protection systems. The K-factor is commonly used in hydraulic calculations for pipe fittings, accessories, sprinklers, and other mechanical equipment. This parameter is associated with various Revit categories, including pipe fittings, pipe accessories, sprinklers, generic models, flexible pipe curves, mechanical equipment, and fire alarm devices.

```
[Guid("48b9c6f5-70b6-4f7f-bd23-5ae171a78a7f")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] DischargeType =
44
```

Represents the discharge type for mechanical systems in Revit. This parameter is used to categorize and define the type of discharge in mechanical systems, such as pipes, fittings, accessories, and other related components. It is associated with multiple Revit categories, including pipes, pipe fittings, pipe accessories, generic models, flexible pipes, sprinklers, and mechanical equipment.

```
[Guid("322979ea-68ed-4725-b563-2cc023159fa9")]
```

```
[SharedParameterType(SharedParameterType.YesNo)] [RevitCategories] DoNotNumber = 42
```

Represents a shared parameter that determines whether numbering should be applied. This parameter is typically used in Revit to control whether elements such as pipes, pipe fittings, or pipe accessories are assigned numbers. It is associated with the categories `BuiltInCategory.OST_PipeCurves`, `BuiltInCategory.OST_PipeFitting`, and `BuiltInCategory.OST_PipeAccessory`.

```
[Guid("b1af6c8f-8b44-4e5d-ad1d-80deb78bc6b6")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] EndPap = 23
```

Represents the end paper (EndPap) parameter for Revit elements, typically used in piping systems. This parameter is associated with Revit categories such as pipes and pipe fittings. It is defined as a shared parameter of type [Text](#).

```
[Guid("d94545bc-4956-411d-a6bc-9ec96101d350")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories]
```

```
ExportedPackageName = 12
```

Represents the "MBF\_Exported\_PackageName" shared parameter. This shared parameter is applicable to pipe curves, pipe fittings, pipe accessories, generic models, model groups, mechanical equipment, sprinklers, fire alarm devices, and flex pipe curves. It is of type [Text](#).

```
[Guid("83d4381a-bfe9-450e-9364-c44036e81756")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] ExportedPdfName = 14
```

Represents the "MBF\_Exported\_PDFName" shared parameter. This shared parameter is applicable to pipe curves, pipe fittings, pipe accessories, generic models, model groups, mechanical equipment, sprinklers, fire alarm devices, and flex pipe curves. It is of type [Text](#).

```
[Guid("1bf7feba-7bca-40c9-8ca0-8365574c2648")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories]
```

```
FabricationGroupName = 16
```

Represents the "MBF\_FabGroupName" shared parameter. This shared parameter is applicable to pipe curves, pipe fittings, pipe accessories, generic models, model groups, mechanical equipment,

sprinklers, fire alarm devices, and flex pipe curves. It is of type [Text](#).

```
[Guid("15b2c379-8df6-4f2d-8499-9883e0256663")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Finish = 7
```

Represents the "MBF\_Finish" shared parameter. This shared parameter is applicable to pipe curves, pipe fittings, pipe accessories, generic models, model groups, mechanical equipment, sprinklers, fire alarm devices, and flex pipe curves. It is of type [Text](#).

```
[Guid("72f35e3d-3bd7-4169-b3c0-c6485cdd39c4")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] HandTight = 10
```

Represents the "MBF\_HandTight" shared parameter. This shared parameter is applicable to pipe curves, pipe fittings, pipe accessories, generic models, model groups, mechanical equipment, sprinklers, fire alarm devices, and flex pipe curves. It is of type [Text](#).

```
[Guid("8495409d-c110-4685-b548-2f1440fa7307")]
[SharedParameterType(SharedParameterType.YesNo)] [RevitCategories]
IgnoreInCalculation = 3
```

Represents the "MBF\_IgnoreInCalculation" shared parameter. This shared parameter is applicable to pipe fittings, sprinklers, pipe accessories, mechanical equipment, and fire alarm devices. It is of type [YesOrNo](#).

```
[Guid("e87615ee-c2b0-4f93-9194-8ffda612cdf6")]
[SharedParameterType(SharedParameterType.YesNo)] [RevitCategories]
IncludeInCalculation = 2
```

Represents the "MBF\_IncludeInCalcs" shared parameter. This shared parameter is applicable to pipe fittings, sprinklers, pipe accessories, mechanical equipment, and fire alarm devices. It is of type [YesOrNo](#).

```
[Guid("997C72F0-F375-439E-9AD1-BC2570651E42")]
[SharedParameterType(SharedParameterType.YesNo)] [RevitCategories] IsFitting = 19
```

Represents the "MBF\_IsFitting" shared parameter. This shared parameter is used for flex pipe curves and is of type [YesOrNo](#).

```
[Guid("a47d1d69-27e0-4012-bb42-64e149bf061a")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] KFactor = 46
```

Represents the K-factor parameter used in mechanical and fire protection systems. The K-factor is commonly used in hydraulic calculations for pipe fittings, accessories, sprinklers, and other mechanical equipment. This parameter is associated with various Revit categories, including pipe fittings, pipe

accessories, sprinklers, generic models, flexible pipe curves, mechanical equipment, and fire alarm devices.

```
[Guid("b06ebb90-4f1e-4881-bfb5-fef6f7a64909")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] MinimumFlow = 5
```

Represents the "MBF\_Min\_Flow" shared parameter. This shared parameter is applicable to pipe fittings, sprinklers, and pipe accessories. It is of type [Text](#).

```
[Guid("467a1334-8e46-4d4d-bf22-0bd377ff6be5")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] MinimumPressure = 4
```

Represents the "MBF\_Min\_Pressure" shared parameter. This shared parameter is applicable to pipe fittings, sprinklers, and pipe accessories. It is of type [Text](#).

```
[Guid("c0932a83-87c2-4286-9c28-d4eb4ae03ef8")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Pap1 = 25
```

Represents a shared parameter type for Revit elements, specifically associated with pipe curves. This class is identified by the GUID "c0932a83-87c2-4286-9c28-d4eb4ae03ef8" and is categorized under the Revit built-in category `BuiltInCategory.OST_PipeCurves`. It is intended for use with shared parameters of type [Text](#).

```
[Guid("64f2f492-72f6-4e92-9b54-4e6bd8845275")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Pap2 = 27
```

Represents a shared parameter with the identifier "2\_PAP" for use in Revit's pipe curves category. This parameter is associated with the `BuiltInCategory.OST_PipeCurves` category in Revit and is defined as a text-based shared parameter. It is identified by the GUID "64f2f492-72f6-4e92-9b54-4e6bd8845275".

```
[Guid("4acd2fa4-e8c4-4d00-89e6-bdeed126fb39")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Pap3 = 29
```

Represents a shared parameter with the identifier "3\_PAP" for use in Revit's pipe curves category. This parameter is associated with the `BuiltInCategory.OST_PipeCurves` category in Revit and is defined as a text-based shared parameter. It is identified by the GUID "4acd2fa4-e8c4-4d00-89e6-bdeed126fb39".

```
[Guid("b710b589-43a3-4663-9f5d-c7c255095194")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Pap4 = 31
```

Represents a shared parameter type with the identifier "4\_PAP" for use in Revit. This parameter is associated with the Revit category `BuiltInCategory.OST_PipeCurves` and is defined as a text-based shared parameter. It is identified by the GUID "b710b589-43a3-4663-9f5d-c7c255095194".

```
[Guid("2073ad09-f450-4369-b152-366f3c97f352")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Pap5 = 33
```

Represents a shared parameter with the identifier "5\_PAP" for use in Revit projects. This parameter is associated with the Revit category `BuiltInCategory.OST_PipeCurves` and is defined as a text-based shared parameter. It is identified by the GUID "2073ad09-f450-4369-b152-366f3c97f352".

```
[Guid("b25b7441-fe65-4a68-b5da-0195cf452a66")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Pap6 = 35
```

Represents a shared parameter with the identifier "6\_PAP" for use in Revit projects. This parameter is associated with the Revit category `BuiltInCategory.OST_PipeCurves` and is defined as a text-based shared parameter. It is identified by the GUID "b25b7441-fe65-4a68-b5da-0195cf452a66".

```
[Guid("38005670-ec77-4a15-a561-ff65273fc09a")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Pap7 = 37
```

Represents a shared parameter with the identifier "7\_PAP" for use in Revit projects. This parameter is associated with the Revit category `BuiltInCategory.OST_PipeCurves` and is defined as a text-based shared parameter. It is identified by the GUID "38005670-ec77-4a15-a561-ff65273fc09a".

```
[Guid("9d447d9e-3100-45b6-9cc9-f6cbe40381ca")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Pap8 = 39
```

Represents a shared parameter with the identifier "8\_PAP" for use in Revit projects. This parameter is associated with pipe curves in Revit and is categorized under `BuiltInCategory.OST_PipeCurves`. It is defined as a text-based shared parameter.

```
[Guid("2b164db7-983f-455e-9b0a-ba77061456d6")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Pap9 = 41
```

Represents a shared parameter with the identifier "9\_PAP" for use in Revit. This parameter is associated with the Revit category `BuiltInCategory.OST_PipeCurves` and is defined as a text-based shared parameter. It is identified by the GUID "2b164db7-983f-455e-9b0a-ba77061456d6".

```
[Guid("c4347a82-7375-42aa-ad1a-96ba6212f122")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] PipeCode = 8
```

Represents the "MBF\_PipeCode" shared parameter. This shared parameter is applicable to pipe curves, pipe fittings, pipe accessories, generic models, model groups, mechanical equipment, sprinklers, fire alarm devices, and flex pipe curves. It is of type [Text](#).

```
[Guid("a137e596-b255-494a-9c8a-1f32f4ec0a5f")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] PipeNoListing =
```

Represents a shared parameter for specifying a pipe number listing in a Revit model. This parameter is associated with Revit categories such as `BuiltInCategory.OST_PipeCurves` and `BuiltInCategory.OST_PipeFitting`. It is defined as a text-based shared parameter and is identified by the GUID "a137e596-b255-494a-9c8a-1f32f4ec0a5f".

```
[Guid("2e83bba8-8d6e-4a90-a29f-108c705a7661")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] ProtectionArea =
6
```

Represents the "MBF\_ProtectionArea" shared parameter. This shared parameter is applicable to pipe fittings, sprinklers, and pipe accessories. It is of type [Text](#).

```
[Guid("5f2c6451-0d24-4421-b3f8-02386c2cd60b")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] RealGroupSize =
18
```

Represents the "MBF\_Allowance" shared parameter. This shared parameter is applicable to flex pipe curves. It is of type [Text](#).

```
[Guid("5f2c6451-0d24-4421-b3f8-02386c2cd60b")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] RealFittingSizes =
13
```

Represents the "MBF\_RealFittingSizes" shared parameter. This shared parameter is applicable to pipe curves, pipe fittings, pipe accessories, generic models, model groups, mechanical equipment, sprinklers, fire alarm devices, and flex pipe curves. It is of type [Text](#).

```
[Guid("5fa85912-77b2-4d48-a7df-5e524aa45c43")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Run1Dim = 24
```

Represents a shared parameter for specifying the dimension of a pipe run in Revit. This parameter is associated with the "1\_RUN DIM" description and is categorized under `BuiltInCategory.OST_PipeCurves`. It is defined as a text-based shared parameter.

```
[Guid("28aba22e-1438-4608-b8ce-a85e52854267")]
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Run2Dim = 26
```

Represents a shared parameter for specifying the 2\_RUN DIM value in Revit. This parameter is associated with pipe curves in Revit and is categorized as a text parameter. It is identified by the GUID "28aba22e-1438-4608-b8ce-a85e52854267".

```
[Guid("47e8420e-dc38-47ce-9dbd-74358a69a28c")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Run3Dim = 28
```

Represents a shared parameter for the "3\_RUN DIM" property in Revit, associated with pipe curves. This parameter is identified by the GUID "47e8420e-dc38-47ce-9dbd-74358a69a28c" and is categorized under BuiltInCategory.OST\_PipeCurves. It is defined as a text-based shared parameter.

```
[Guid("216fb144-f402-4d1b-b6f9-dc1f62850017")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Run4Dim = 30
```

Represents a shared parameter for specifying the "4\_RUN DIM" value in Revit pipe curves. This parameter is associated with the "4\_RUN DIM" description and is categorized under BuiltInCategory.OST\_PipeCurves . It is defined as a text-based shared parameter.

```
[Guid("236886e5-b9d9-4a7a-a456-27324bb33869")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Run5Dim = 32
```

Represents a shared parameter for specifying the "5\_RUN DIM" value in Revit. This parameter is associated with the "PipeCurves" category in Revit and is of type [Text](#) . It is identified by the GUID "236886e5-b9d9-4a7a-a456-27324bb33869".

```
[Guid("5914c51d-8028-48c3-bde0-98e21dca0d67")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Run6Dim = 34
```

Represents a shared parameter for specifying the "6\_RUN DIM" value associated with pipe curves in Revit. This parameter is categorized under the Revit built-in category BuiltInCategory.OST\_PipeCurves and is defined as a text-based shared parameter. It is identified by the GUID "5914c51d-8028-48c3-bde0-98e21dca0d67".

```
[Guid("fc237068-ae9a-45eb-81da-f4a3a7c00a38")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Run7Dim = 36
```

Represents a shared parameter for specifying the "7\_RUN DIM" value in Revit. This parameter is associated with the "PipeCurves" category in Revit and is of type [Text](#) . It is identified by the GUID "fc237068-ae9a-45eb-81da-f4a3a7c00a38".

```
[Guid("62244904-01e1-45f3-a2c6-cf15ea715713")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Run8Dim = 38
```

Represents a shared parameter for specifying the "8\_RUN DIM" value in Revit pipe curves. This parameter is associated with the "8\_RUN DIM" description and is categorized under BuiltInCategory.OST\_PipeCurves . It is defined as a text-based shared parameter.

```
[Guid("a5f94330-0a3b-4242-91be-8a66ddc62e56")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] Run9Dim = 40
```

Represents a shared parameter for the "9\_RUN DIM" dimension, associated with pipe curves in Revit. This parameter is categorized under the BuiltInCategory.OST\_PipeCurves category and is defined as a length parameter. It is identified by the GUID "a5f94330-0a3b-4242-91be-8a66ddc62e56".

```
[Guid("ab58b0c7-0375-4ead-a18f-27f84379fa19")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] SprinklerGroupName = 45
```

Represents the discharge type for mechanical systems in Revit. This parameter is used to categorize and define the type of discharge in mechanical systems, such as pipes, fittings, accessories, and other related components. It is associated with multiple Revit categories, including pipes, pipe fittings, pipe accessories, generic models, flexible pipes, sprinklers, and mechanical equipment.

```
[Guid("ab1be1ed-1730-433c-b2c6-51edfaf84b5")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] StartPap = 22
```

Represents the "Start Pap" shared parameter, which is associated with pipe curves and pipe fittings in Revit. This parameter is categorized under BuiltInCategory.OST\_PipeCurves and BuiltInCategory.OST\_PipeFitting . It is defined as a text-based shared parameter and is identified by the GUID "ab1be1ed-1730-433c-b2c6-51edfaf84b5".

```
[Guid("9d698cc7-3c6d-466e-9f89-ee1baf1326a2")]
```

```
[SharedParameterType(SharedParameterType.Text)] [RevitCategories] SubType = 15
```

Represents the "MBF\_Subtype" shared parameter. This shared parameter is applicable to pipe curves, pipe fittings, pipe accessories, generic models, model groups, mechanical equipment, sprinklers, fire alarm devices, and flex pipe curves. It is of type [Text](#).

# Namespace MB.Common.Tools

## Classes

### [MbftoolExtensions](#)

Provides extension methods for the [Mbftool](#) class.

## Enums

### [Mbftool](#)

Enum representing various tools in the MBF (MicroBIM Fire) suite. Each tool is decorated with descriptive metadata and optional shared parameter requirements.

# Enum Mbftool

Namespace: [MB.Common.Tools](#)

Assembly: MB.Common.dll

Enum representing various tools in the MBF (MicroBIM Fire) suite. Each tool is decorated with descriptive metadata and optional shared parameter requirements.

```
public enum Mbftool
```

## Extension Methods

[MbftoolExtensions.GetRequiredSharedParameters\(Mbftool\)](#)

## Fields

**ApplyParameters = 4**

Represents the Apply Parameters tool.

**AssemblyManager = 2**

Represents the Assembly Manager tool.

**AutoSprinklerSpacing = 6**

Represents the Auto Sprinkler Spacing tool.

**CutLengthPipeOptimization = 3**

Represents the Cut Length Pipe Optimization tool.

```
[RequiredSharedParameters(new ToolsSharedParameter[] { ToolsSharedParameter.Color,
ToolsSharedParameter.Color, ToolsSharedParameter.PipeCode,
ToolsSharedParameter.AddFitToId, ToolsSharedParameter.HandTight,
ToolsSharedParameter.DescriptionAndAbbreviationListNo,
ToolsSharedParameter.ExportedPackageName, ToolsSharedParameter.RealFittingSizes,
ToolsSharedParameter.ExportedPdfName, ToolsSharedParameter.SubType,
ToolsSharedParameter.FabricationGroupName, ToolsSharedParameter.Allowance,
ToolsSharedParameter.RealGroupSize, ToolsSharedParameter.IsFitting,
ToolsSharedParameter.CalculateSum })] Mapper = 5
```

Represents the Mapper tool, used to assign various metadata and fabrication-related properties. This tool requires several shared parameters including: [Color](#), [Finish](#), [PipeCode](#), and many others used for downstream fabrication and reporting.

```
[RequiredSharedParameters(new ToolsSharedParameter[] {  
    ToolsSharedParameter.ActualCoverage, ToolsSharedParameter.ProtectionArea })]  
SprinklerVoronoi = 1
```

Represents the Sprinkler Voronoi tool. This tool requires the [ActualCoverage](#) and [ProtectionArea](#) shared parameters.

## Remarks

Tools are annotated with [DescriptionAttribute](#) for display purposes, and with [RequiredSharedParametersAttribute](#) to define necessary shared parameters.

# Class MbftoolExtensions

Namespace: [MB.Common.Tools](#)

Assembly: MB.Common.dll

Provides extension methods for the [Mbftool](#) class.

```
public static class MbftoolExtensions
```

## Inheritance

[object](#) ← MbftoolExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### GetRequiredSharedParameters(Mbftool)

Retrieves the required shared parameters associated with the specified [Mbftool](#).

```
public static ToolsSharedParameter[] GetRequiredSharedParameters(this Mbftool tool)
```

#### Parameters

tool [Mbftool](#)

The [Mbftool](#) instance for which the required shared parameters are retrieved.

#### Returns

[ToolsSharedParameter\[\]](#)

An array of [ToolsSharedParameter](#) objects if the [RequiredSharedParametersAttribute](#) is found; otherwise, an empty array.

# Namespace MB.Common.Units.Behaviours

## Classes

### [TextBoxUnitLengthConversionBehavior](#)

Represents a behavior that can be attached to a MicroBIM.UI.Controls.TextBox to enable or disable unit length conversion functionality when a specific key is pressed.

# Class TextBoxUnitLengthConversionBehavior

Namespace: [MB.Common.Units.Behaviours](#)

Assembly: MB.Common.dll

Represents a behavior that can be attached to a MicroBIM.UI.Controls.TextBox to enable or disable unit length conversion functionality when a specific key is pressed.

```
public class TextBoxUnitLengthConversionBehavior : Behavior<TextBox>,  
IAnimatable, IAttachedObject
```

## Inheritance

```
object ↪ ← DispatcherObject ↪ ← DependencyObject ↪ ← Freezable ↪ ← Animatable ↪ ← Behavior ←  
Behavior<TextBox> ← TextBoxUnitLengthConversionBehavior
```

## Implements

[IAnimatable](#), [IAttachedObject](#)

## Inherited Members

```
Behavior<TextBox>.AssociatedObject , Behavior.Attach\(DependencyObject\) , Behavior.Detach() ,  
Behavior.CreateInstanceCore() , Behavior.AssociatedType , Animatable.Clone\(\) ,  
Animatable.FreezeCore\(bool\) ,  
Animatable.ApplyAnimationClock\(DependencyProperty, AnimationClock\) ,  
Animatable.ApplyAnimationClock\(DependencyProperty, AnimationClock, HandoffBehavior\) ,  
Animatable.BeginAnimation\(DependencyProperty, AnimationTimeline\) ,  
Animatable.BeginAnimation\(DependencyProperty, AnimationTimeline, HandoffBehavior\) ,  
Animatable.GetAnimationBaseValue\(DependencyProperty\) , Animatable.HasAnimatedProperties ,  
Freezable.CloneCurrentValue\(\) , Freezable.GetAsFrozen\(\) , Freezable.GetCurrentValueAsFrozen\(\) ,  
Freezable.Freeze\(\) , Freezable.OnPropertyChanged\(DependencyPropertyChangedEventArgs\) ,  
Freezable.CreateInstance\(\) , Freezable.CloneCore\(Freezable\) ,  
Freezable.CloneCurrentValueCore\(Freezable\) , Freezable.GetAsFrozenCore\(Freezable\) ,  
Freezable.GetCurrentValueAsFrozenCore\(Freezable\) , Freezable.OnChanged\(\) ,  
Freezable.ReadPreamble\(\) , Freezable.WritePreamble\(\) , Freezable.WritePostscript\(\) ,  
Freezable.OnFreezablePropertyChanged\(DependencyObject, DependencyObject\) ,  
Freezable.OnFreezablePropertyChanged\(DependencyObject, DependencyObject, DependencyProperty\) ,  
Freezable.Freeze\(Freezable, bool\) , Freezable.CanFreeze , Freezable.IsFrozen , Freezable.Changed ,  
DependencyObject.Equals\(object\) , DependencyObject.GetHashCode\(\) ,  
DependencyObject.GetValue\(DependencyProperty\) ,  
DependencyObject.SetValue\(DependencyProperty, object\) ,
```

[DependencyObject.SetCurrentValue\(DependencyProperty, object\)](#) ,  
[DependencyObject.SetValue\(DependencyPropertyKey, object\)](#) ,  
[DependencyObject.ClearValue\(DependencyProperty\)](#) ,  
[DependencyObject.ClearValue\(DependencyPropertyKey\)](#) ,  
[DependencyObject.CoerceValue\(DependencyProperty\)](#) ,  
[DependencyObject.InvalidateProperty\(DependencyProperty\)](#) ,  
[DependencyObject.ShouldSerializeProperty\(DependencyProperty\)](#) ,  
[DependencyObject.ReadLocalValue\(DependencyProperty\)](#) ,  
[DependencyObject.GetLocalValueEnumerator\(\)](#) , [DependencyObject.DependencyObjectType](#) ,  
[DependencyObject.IsSealed](#) , [DispatcherObject.Dispatcher](#) , [object.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This behavior listens for key events on the associated MicroBIM.UI.Controls.TextBox and applies a group size conversion when the Enter key is pressed, provided that the [EnableGroupSizeConversion](#) property is set to [true](#).

## Constructors

### TextBoxUnitLengthConversionBehavior()

```
public TextBoxUnitLengthConversionBehavior()
```

## Fields

### EnableGroupSizeConversionProperty

Identifies the [EnableGroupSizeConversion](#) dependency property.

```
public static readonly DependencyProperty EnableGroupSizeConversionProperty
```

## Field Value

[DependencyProperty](#)

## Remarks

This property determines whether the group size conversion feature is enabled for the associated TextBoxUnitLengthConversionBehavior. The default value is [false](#).

## Properties

### EnableGroupSizeConversion

Gets or sets a value indicating whether group size conversion is enabled.

```
public bool EnableGroupSizeConversion { get; set; }
```

#### Property Value

[bool](#)

## Methods

### OnAttached()

Attaches event handlers and initializes behavior when the behavior is attached to an object.

```
protected override void OnAttached()
```

#### Remarks

If [EnableGroupSizeConversion](#) is [true](#), subscribes to the [KeyDown](#) event of the associated object to handle key press events.

### OnDetaching()

Detaches the behavior from the associated object, removing any event handlers.

```
protected override void OnDetaching()
```

#### Remarks

This method is called when the behavior is being detached from its associated object. It removes the `MB.Common.Units.Behaviours.TextBoxUnitLengthConversionBehavior.OnKeyDown(System.Object, System.Windows.Input.KeyEventArgs)` event handler from the `AssociatedObject`.

# Namespace MB.Common.Units.Converters

## Classes

### [RevitLengthValueConverter](#)

Provides methods to convert length values between internal units and formatted strings suitable for display in a Revit application.

# Class RevitLengthValueConverter

Namespace: [MB.Common.Units.Converters](#)

Assembly: MB.Common.dll

Provides methods to convert length values between internal units and formatted strings suitable for display in a Revit application.

```
public class RevitLengthValueConverter : IValueConverter
```

## Inheritance

[object](#) ← RevitLengthValueConverter

## Implements

[IValueConverter](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This converter is designed to work with Revit's unit system, allowing for the conversion of length values to and from their string representations based on the active document's unit settings. It supports both forward and backward conversion, handling various input types such as [double](#) and [string](#).

## Constructors

### RevitLengthValueConverter()

```
public RevitLengthValueConverter()
```

## Methods

### Convert(object, Type, object, CultureInfo)

Converts a given value to a formatted string representation based on the specified target type and culture.

```
public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
```

## Parameters

**value** [object](#)

The value to convert, which can be a double or a string representing a numeric value.

**targetType** [Type](#)

The type to which the value is being converted. This parameter is not used in the conversion process.

**parameter** [object](#)

An optional parameter for the conversion. This parameter is not used in the conversion process.

**culture** [CultureInfo](#)

The culture information to use during conversion, affecting number formatting.

## Returns

[object](#)

A string representing the formatted value with unit symbols appended, or the original value if it cannot be converted.

## Remarks

The method formats numeric values according to the active document's unit settings and appends unit symbols. If the input value is a string, it attempts to parse it as a numeric value before formatting.

## ConvertBack(object, Type, object, CultureInfo)

Converts a value back to its original type, interpreting strings with units or converting double values from internal units.

```
public object ConvertBack(object value, Type targetType, object parameter,
CultureInfo culture)
```

## Parameters

### **value** [object](#)

The value to convert back. This can be a string representing a value with units or a double representing a length in internal units.

### **targetType** [Type](#)

The type to which the value is being converted. This parameter is not used in the conversion process.

### **parameter** [object](#)

An optional parameter for the conversion. This parameter is not used in the conversion process.

### **culture** [CultureInfo](#)

The culture to use in the conversion process. This parameter is not used in the conversion process.

## Returns

### [object](#)

The converted value as a double if the input is a valid string with units or a double in internal units; otherwise, returns [DoNothing](#).

## Remarks

This method supports conversion of string inputs that include units and double values representing lengths in internal units. If the input is a string, it attempts to parse it as a value with units. If the input is a double, it converts it from internal units using the current document's unit settings. If the conversion cannot be performed, [DoNothing](#) is returned.

## TryParseValueWithUnits(string, ForgeTypeId, out double)

Attempts to parse a string input representing a value with units into a double, based on the specified unit type.

```
public static bool TryParseValueWithUnits(string input, ForgeTypeId specTypeId, out  
double internalValue)
```

## Parameters

## **input** [string](#)

The input string containing the value and units to parse. This string is case-insensitive and will be trimmed of whitespace.

## **specTypeId** [ForgeTypeld](#)

The unit type identifier used to interpret the units in the input string.

## **internalValue** [double](#)

When this method returns, contains the parsed double value if the conversion succeeded, or zero if it failed.

## Returns

## [bool](#)

[true](#) if the input string was successfully parsed into a double value; otherwise, [false](#).

## Remarks

This method uses the active document's units to interpret the input string. If the active document or its units are unavailable, the method returns [false](#).

# Namespace MB.Common.Units.Helpers

## Classes

### [TextBoxLengthConversionHelper](#)

Provides utility methods for converting and formatting text box input values related to length measurements.

# Class TextBoxLengthConversionHelper

Namespace: [MB.Common.Units.Helpers](#)

Assembly: MB.Common.dll

Provides utility methods for converting and formatting text box input values related to length measurements.

```
public static class TextBoxLengthConversionHelper
```

## Inheritance

[object](#) ← TextBoxLengthConversionHelper

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This helper class includes methods to apply group size conversions to text box inputs and to convert numeric values to the current unit system with appropriate formatting. It is designed to work with different versions of the API, handling unit specifications and formatting options accordingly.

## Methods

### ApplyGroupSizeConversion(TextBox)

Converts the text in the specified MicroBIM.UI.Controls.TextBox to a standardized group size format.

```
public static void ApplyGroupSizeConversion(TextBox textBox)
```

#### Parameters

**textBox** TextBox

The MicroBIM.UI.Controls.TextBox whose text will be converted.

#### Remarks

This method updates the text of the provided MicroBIM.UI.Controls.TextBox by applying a conversion to ensure the text represents a valid group size format. The conversion logic is defined within the method and affects the text directly.

## ConvertTextBoxGroupSize(string)

Converts a string representation of a length value into a formatted string with unit symbols.

```
public static string ConvertTextBoxGroupSize(string value)
```

Parameters

**value** [string](#)

The string representation of the length value to convert.

Returns

[string](#)

A formatted string that includes the length value with unit symbols appended.

Remarks

The method uses the active document's unit settings to parse and format the length value. It ensures that the formatted output includes unit symbols and suppresses trailing zeros.

## ConvertToCurrentUnits(double)

Converts a numeric value to a string representation in the current document's units.

```
public static string ConvertToCurrentUnits(double value)
```

Parameters

**value** [double](#)

The numeric value to convert.

Returns

## string ↗

A string representing the value formatted according to the current document's unit settings, including unit symbols. If the document's units are not available, returns the value formatted to two decimal places.

### Remarks

The method adapts to different versions of the API, ensuring compatibility with both older and newer versions. It applies formatting options such as suppressing trailing and leading zeros and appending unit symbols to the formatted string.

# Namespace MB.Common.Utils.Attributes

## Classes

### [AttributeUtils](#)

Provides utility methods for working with custom attributes in .NET.

# Class AttributeUtils

Namespace: [MB.Common.Utils.Attributes](#)

Assembly: MB.Common.dll

Provides utility methods for working with custom attributes in .NET.

```
public static class AttributeUtils
```

## Inheritance

[object](#) ← AttributeUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## GetAttribute<T>(object)

Retrieves a custom attribute of the specified type **T** applied to a given enum value or field.

```
public static T GetAttribute<T>(object value) where T : Attribute
```

### Parameters

**value** [object](#)

The target enum value or field.

### Returns

**T**

The attribute of type **T** if it is applied; otherwise, throws an exception.

### Type Parameters

**T**

The type of the attribute to retrieve.

## GetFieldAttribute<T, TType>(string)

Retrieves a custom attribute of the specified type **T** applied to a given field in a class or struct.

```
public static T GetFieldAttribute<T, TType>(string fieldName) where T : Attribute
```

### Parameters

**fieldName** [string](#)

The name of the field in the class or struct.

### Returns

**T**

The attribute of type **T** if it is applied; otherwise, throws an exception.

### Type Parameters

**T**

The type of the attribute to retrieve.

**TType**

The type of the target class or struct.

## TryGetAttribute<T>(object, out T)

Attempts to retrieve a custom attribute of the specified type **T** applied to a given enum value or field without throwing an exception if it is not found.

```
public static bool TryGetAttribute<T>(object value, out T attribute) where T : Attribute
```

### Parameters

**value** [object](#)

The target enum value or field.

## attribute T

When this method returns, contains the attribute of type `T` if found; otherwise, `null`.

Returns

`bool` ↗

True if the attribute is found; otherwise, false.

Type Parameters

`T`

The type of the attribute to retrieve.

# Namespace MB.Common.Utils.Handlers

## Classes

### [MbHandlers](#)

Provides static access to Revit external event handlers used across the application.

# Class MbHandlers

Namespace: [MB.Common.Utils.Handlers](#)

Assembly: MB.Common.dll

Provides static access to Revit external event handlers used across the application.

```
public static class MbHandlers
```

## Inheritance

[object](#) ← MbHandlers

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This utility class simplifies the management of Revit's Nice3point.Revit.Toolkit.External.Handlers.Action EventHandler and Nice3point.Revit.Toolkit.External.Handlers.AsyncEventHandler instances, allowing them to be globally registered and reused.

## Properties

### AsyncExternalEventHandler

Gets or sets the asynchronous external event handler for executing asynchronous Revit commands.

```
public static AsyncEventHandler AsyncExternalEventHandler { get; }
```

#### Property Value

AsyncEventHandler

### ExternalEventHandler

Gets or sets the synchronous external event handler for executing Revit commands.

```
public static ActionEventHandler ExternalEventHandler { get; }
```

Property Value

ActionEventHandler

## Methods

### RegisterHandlers()

Registers and initializes the external event handlers used by the application. Call this method during application startup or initialization.

```
public static void RegisterHandlers()
```

# Namespace MB.Common.Utils.Image

## Classes

### [ImageUtils](#)

Provides utility functions for working with images, such as resizing with high-quality settings.

# Class ImageUtils

Namespace: [MB.Common.Utils.Image](#)

Assembly: MB.Common.dll

Provides utility functions for working with images, such as resizing with high-quality settings.

```
public static class ImageUtils
```

## Inheritance

[object](#) ← ImageUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### ResizeImage(Image, int, int)

Resizes an image to the specified dimensions while preserving visual quality using advanced graphics settings.

```
public static Bitmap ResizeImage(Image image, int width, int height)
```

#### Parameters

**image** [Image](#)

The source [Image](#) to be resized.

**width** [int](#)

The target width of the resized image.

**height** [int](#)

The target height of the resized image.

## Returns

[Bitmap](#)

A new [Bitmap](#) instance containing the resized image.

# Namespace MB.Common.Utils.Loaders

## Classes

### [MbResourceLoader](#)

Provides utility methods for working with embedded resources, including loading JSON content and extracting files.

# Class MbResourceLoader

Namespace: [MB.Common.Utils.Loaders](#)

Assembly: MB.Common.dll

Provides utility methods for working with embedded resources, including loading JSON content and extracting files.

```
public static class MbResourceLoader
```

## Inheritance

[object](#) ← MbResourceLoader

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## EnsureFileFromEmbeddedResource(string, string)

Ensures that a file exists at the specified path. If it does not, the method creates it using the content from an embedded resource.

```
public static MbResult<bool> EnsureFileFromEmbeddedResource(string outputPath,  
    string embeddedResourceName)
```

### Parameters

**outputPath** [string](#)

The full path where the file should be created if it doesn't exist.

**embeddedResourceName** [string](#)

The full name of the embedded resource (e.g., "Namespace.Resources.Default.json").

### Returns

## [MbResult<bool>](#)

An [MbResult<T>](#) indicating the outcome:

- `Success = true` — The file already exists or was created successfully from the embedded resource.
- `Success = false` — The resource was not found or an error occurred during file creation. The `ErrorMessage` will describe the issue.

## Remarks

This method can be used to seed configuration files or default templates on first run of the application. It supports nested directory creation and uses the currently executing assembly to locate the resource.

## LoadEmbeddedJsonAsync<T>(string, Assembly)

Loads a JSON list of objects from an embedded resource using Newtonsoft.Json.

```
public static Task<List<T>> LoadEmbeddedJsonAsync<T>(string resourceName,  
Assembly assembly)
```

## Parameters

`resourceFileName` [string](#)

The name of the embedded resource file.

`assembly` [Assembly](#)

The assembly containing the embedded resource.

## Returns

[Task](#)<[List](#)<T>>

A list of deserialized objects, or an empty list on failure.

## Type Parameters

T

The type of objects in the JSON list.

# Namespace MB.Common.Utils.Result

## Classes

### [MbResult<T>](#)

Represents the result of an operation, encapsulating the success status, value, and error message.

# Class MbResult<T>

Namespace: [MB.Common.Utils.Result](#)

Assembly: MB.Common.dll

Represents the result of an operation, encapsulating the success status, value, and error message.

```
public class MbResult<T>
```

## Type Parameters

T

The type of the result value.

## Inheritance

[object](#) ← MbResult<T>

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Properties

## ErrorMessage

Gets the error message if the operation failed.

```
public string ErrorMessage { get; }
```

## Property Value

[string](#)

## IsSuccess

Gets a value indicating whether the operation was successful.

```
public bool IsSuccess { get; }
```

Property Value

[bool](#)

Value

Gets the value of the result if the operation was successful.

```
public T Value { get; }
```

Property Value

T

## Methods

**Failure(string)**

Creates a failed result containing the specified error message.

```
public static MbResult<T> Failure(string errorMessage)
```

Parameters

errorMessage [string](#)

The error message.

Returns

[MbResult](#)<T>

A failed [MbResult](#)<T>.

**Success(T)**

Creates a successful result containing the specified value.

```
public static MbResult<T> Success(T value)
```

## Parameters

**value** T

The result value.

## Returns

[MbResult](#)<T>

A successful [MbResult](#)<T>.

# Namespace MB.Common.ViewLoader

## Classes

### [IsolatedViewLoader](#)

Provides extension methods for loading XAML views manually and in isolation from a URI. This is used as a workaround for known issues in some WPF/Revit XAML loading contexts.

# Class IsolatedViewLoader

Namespace: [MB.Common.ViewLoader](#)

Assembly: MB.Common.dll

Provides extension methods for loading XAML views manually and in isolation from a URI. This is used as a workaround for known issues in some WPF/Revit XAML loading contexts.

```
public static class IsolatedViewLoader
```

## Inheritance

[object](#) ← IsolatedViewLoader

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### LoadViewFromUri(FrameworkElement, string)

Loads a XAML view from a given base URI into a [FrameworkElement](#). Uses reflection to bypass internal WPF methods and load XAML manually.

```
public static void LoadViewFromUri(this FrameworkElement view, string baseUri)
```

#### Parameters

**view** [FrameworkElement](#)

The view instance into which the XAML will be loaded.

**baseUri** [string](#)

The relative URI to the XAML file (e.g. "/AssemblyName;component/ViewFolder/View.xaml").

### LoadView<T>(T)

Loads the corresponding XAML view based on the type of the view instance. Infers the URI automatically from the view's type and namespace.

```
public static void LoadView<T>(this T view) where T : FrameworkElement
```

## Parameters

**view** **T**

The view instance to load.

## Type Parameters

**T**

The type of the view, must inherit from [FrameworkElement](#).

# Namespace MBF.Revit.Creation.Constants

## Enums

### [SpecialReferenceType](#)

Defines a set of reference types that represent various alignment, directional, positional, and layout options.

# Enum SpecialReferenceType

Namespace: [MBF.Revit.Creation.Constants](#)

Assembly: MBF.Revit.Creation.dll

Defines a set of reference types that represent various alignment, directional, positional, and layout options.

```
public enum SpecialReferenceType
```

## Fields

**Back = 5**

Represents the "Back" direction in an enumeration of possible directions.

**Bottom = 6**

Represents the bottom alignment option in a layout or positioning context.

**CenterElevation = 7**

Represents the elevation of the center point in a specific context.

**CenterLeftOrRight = 1**

Represents an alignment option where content is centered horizontally.

**CentreFrontOrBack = 4**

Represents the center fullback position in a sports team lineup.

**ConnectorPoint = 3001**

Represents the identifier for a connector point in a graphical or diagramming context. This value is typically used to identify and differentiate connector points within a system. It can be used in scenarios such as diagramming tools, graphical editors, or other applications that involve connecting elements.

**Front = 3**

Represents the front side of an object or entity.

**HorizontalTop = 42**

Represents a horizontal alignment positioned at the top.

**Inclined = 43**

Represents an inclined orientation with a specific value of 43.

**Left = 0**

Represents the left alignment or direction in a layout or operation.

**NewRef1 = 10**

Represents a constant value used for a specific purpose in the application. This value is set to 10 and is intended to be used as a reference or default value in scenarios where such a constant is required. Ensure that its usage aligns with the intended context.

**NewRef2 = 11**

Represents a constant value of 11.

**NewRef3 = 12**

Represents a constant value used for a specific purpose in the application. The value of **newref3** is set to 12. Ensure that this constant is used in contexts where this specific value is required.

**NewRef4 = 13**

Represents a constant value of 13.

**NewRef5 = 14**

Represents a constant value of 14.

**NewRef6 = 15**

Represents a constant value of 15.

**Right = 2**

Represents the right direction in a directional enumeration.

**Top = 8**

Represents the top alignment option in a layout or positioning context.

**VerticalStack1 = 39**

Represents a vertical stack layout with a predefined configuration.

#### **VerticalStack3 = 40**

Represents a vertical stack layout with three elements, identified by the value 40. This value can be used to specify a predefined layout configuration for UI components.

#### **VerticalStack4 = 41**

Represents a vertical stack layout with four elements.

## **Remarks**

This enumeration provides a comprehensive list of constants that can be used to specify alignment, direction, layout configurations, or other contextual references in applications. Each value represents a specific concept, such as alignment (e.g., [Left](#), [Right](#)), layout configurations (e.g., [VerticalStack1](#), [HorizontalTop](#)), or predefined constants (e.g., [NewRef1](#), [ConnectorPoint](#)).

# Namespace MBF.Revit.Creation.Extensions

## Classes

### [DocumentExtensions](#)

Provides extension methods for working with Revit documents, specifically for managing linked Revit models.

# Class DocumentExtensions

Namespace: [MBF.Revit.Creation.Extensions](#)

Assembly: MBF.Revit.Creation.dll

Provides extension methods for working with Revit documents, specifically for managing linked Revit models.

```
public static class DocumentExtensions
```

## Inheritance

[object](#) ← DocumentExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes methods to retrieve and manipulate linked Revit model instances within a document. These methods are designed to simplify common tasks related to linked models, such as retrieving all valid linked models or ensuring that linked models are set to be room bounding.

## Methods

### GetAllLinkedRevitModels(Document)

Retrieves all valid linked Revit model instances in the document.

```
public static List<RevitLinkInstance> GetAllLinkedRevitModels(this Document doc)
```

#### Parameters

**doc** Document

The Revit document to search in.

#### Returns

## List <RevitLinkInstance>

A list of Autodesk.Revit.DB.RevitLinkInstance objects representing valid linked models.

## MakeLinkedModelsRoomBounding(Document)

Ensures that all linked Revit models in the document are set to be room bounding.

```
public static void MakeLinkedModelsRoomBounding(this Document doc)
```

### Parameters

**doc** Document

The Revit document in which to update linked model settings.

# Namespace MBF.Revit.Creation.Models

## Classes

### [CategorizedReferences](#)

Data structure to hold categorized references.

# Class CategorizedReferences

Namespace: [MBF.Revit.Creation.Models](#)

Assembly: MBF.Revit.Creation.dll

Data structure to hold categorized references.

```
public class CategorizedReferences
```

## Inheritance

[object](#) ← CategorizedReferences

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### CategorizedReferences()

```
public CategorizedReferences()
```

## Properties

### LineReferences

Gets or sets the collection of line references associated with the current object.

```
public List<Reference> LineReferences { get; set; }
```

Property Value

[List](#) <Reference>

### PlaneReferences

Gets or sets the collection of plane references.

```
public List<Reference> PlaneReferences { get; set; }
```

Property Value

[List](#) <Reference>

## PointReferences

Gets or sets the collection of point references associated with this instance.

```
public List<Reference> PointReferences { get; set; }
```

Property Value

[List](#) <Reference>

# Namespace MBF.Revit.Creation.Utils

## Classes

### [MbDimensionUtils](#)

Provides utility methods for working with special references in Revit family instances.

# Class MbDimensionUtils

Namespace: [MBF.Revit.Creation.Utils](#)

Assembly: MBF.Revit.Creation.dll

Provides utility methods for working with special references in Revit family instances.

```
public static class MbDimensionUtils
```

## Inheritance

[object](#) ← MbDimensionUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes methods for generating, searching, and categorizing stable references within Revit family instances. These utilities are designed to handle complex scenarios such as nested geometry, custom reference tokens, and categorized reference collections.

## Methods

### GetAllSpecialFamilyReferences(FamilyInstance)

Retrieves all special references (plane, line, and point references) from the specified family instance.

```
public static MbResult<CategorizedReferences> GetAllSpecialFamilyReferences(FamilyInstance  
familyInstance)
```

#### Parameters

**familyInstance** FamilyInstance

The FamilyInstance from which to extract references. Must not be [null](#).

#### Returns

## [MbResult<CategorizedReferences>](#)

An [MbResult<T>](#) containing a [CategorizedReferences](#) object with the extracted references if successful; otherwise, a failure result with an error message.

## Remarks

This method categorizes the references into planes, lines, and points by analyzing the geometry of the provided family instance. The method will fail if the family instance is [null](#), if its associated document is unavailable, or if no geometry or valid references are found.

## GetSpecialFamilyReference(FamilyInstance, SpecialReferenceType)

Retrieves a specific reference from a given FamilyInstance based on the specified [SpecialReferenceType](#).

```
public static MbResult<Reference> GetSpecialFamilyReference(FamilyInstance familyInstance, SpecialReferenceType refType)
```

## Parameters

### **familyInstance** FamilyInstance

The FamilyInstance from which to extract the reference. Cannot be [null](#).

### **refType** [SpecialReferenceType](#)

The type of special reference to retrieve, represented by the [SpecialReferenceType](#) enumeration.

## Returns

## [MbResult<Reference>](#)

An [MbResult<T>](#) containing the resolved Autodesk.Revit.DB.Reference if successful, or an error message if the operation fails.

## Remarks

This method processes the geometry of the provided FamilyInstance to construct and resolve a stable reference based on the specified **refType**. The method ensures that the reference is valid and can be used to access the corresponding geometry object. If any step in the process fails, an error message is returned in the result.

## GetSpecialFamilyReference(FamilyInstance, string)

Retrieves a reference to a specific family instance in the document, based on a custom identifier.

```
public static MbResult<Reference> GetSpecialFamilyReference(FamilyInstance familyInstance,  
    string customId)
```

### Parameters

**familyInstance** FamilyInstance

The FamilyInstance from which the reference is to be retrieved. This parameter cannot be [null](#).

**customId** [string](#)

A custom identifier used to build the reference. This parameter cannot be [null](#), empty, or consist only of whitespace.

### Returns

[MbResult](#)<Reference>

An [MbResult](#)<T> containing the Autodesk.Revit.DB.Reference if the operation succeeds, or an error message if the operation fails.

### Remarks

This method performs a series of validation and processing steps to generate a stable reference for the specified family instance. If any step fails, the method returns a failure result with an appropriate error message.

## GetSpecialFamilyReferenceBySearch(FamilyInstance, SpecialReferenceType, string)

Searches for a specific geometry reference within a family instance based on the provided search string and reference type.

```
public static MbResult<Reference> GetSpecialFamilyReferenceBySearch(FamilyInstance  
    familyInstance, SpecialReferenceType refType, string searchString)
```

### Parameters

## **familyInstance** FamilyInstance

The family instance to search within. Must not be [null](#).

## **refType** [SpecialReferenceType](#)

The type of reference to search for, such as a specific geometry or element type.

## **searchString** [string](#)

The string to match against the geometry or element names. Must not be [null](#) or empty.

## Returns

### [MbResult<Reference>](#)

A [MbResult<T>](#) containing the found Autodesk.Revit.DB.Reference if the search is successful; otherwise, a failure result with an appropriate error message.

## Remarks

This method searches both the symbol and instance geometry of the family instance, including nested geometry instances. If no matching reference is found, the method returns a failure result with a descriptive error message.

# Namespace MBF.Revit.Data.Clipper

## Classes

### [CurveLoopProcessor](#)

Processes curve loops and returns chained loops sorted by area. Optionally, the resulting curves can be drawn in the active Revit document.

# Class CurveLoopProcessor

Namespace: [MBF.Revit.Data.Clipper](#)

Assembly: MBF.Revit.Data.dll

Processes curve loops and returns chained loops sorted by area. Optionally, the resulting curves can be drawn in the active Revit document.

```
public static class CurveLoopProcessor
```

## Inheritance

[object](#) ← CurveLoopProcessor

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This method takes a collection of curve loops, filters them based on a minimum area threshold, and sorts them by area in descending order. The resulting loops are returned as a list of chained curves. If `drawCurvesInRevit` is set to [true](#), the processed curves are drawn in the active Revit view.

## Methods

### GetChainedLoops(Document, List<List<Curve>>, bool, double)

Processes a collection of curve loops to identify and return chained loops that meet specified criteria.

```
public static MbResult<List<List<Curve>>> GetChainedLoops(Document doc, List<List<Curve>>  
inputCurves, bool drawCurvesInRevit = true, double minArea = 22500)
```

## Parameters

`doc` Document

The Revit document in which the curves may be drawn if `drawCurvesInRevit` is [true](#).

`inputCurves` [List](#)<[List](#)<Curve>>

A list of curve loops, where each loop is represented as a list of Autodesk.Revit.DB.Curve objects. Each loop must contain at least two curves.

#### drawCurvesInRevit [bool](#)

A value indicating whether the resulting chained loops should be drawn in the Revit document. [true](#) to draw the curves in Revit; otherwise, [false](#).

#### minArea [double](#)

The minimum area threshold for a loop to be included in the result. Loops with an area smaller than this value are excluded. The default value is 150.0 \* 150.0.

## Returns

#### [MbResult](#)<[List](#)<[List](#)<[Curve](#)>>>

An [MbResult<T>](#) containing a list of chained curve loops that meet the specified criteria. If the operation fails, the result contains an error message.

## Remarks

This method performs the following operations:

- Validates the input curve loops to ensure they are non-null and contain at least two curves per loop.
- Uses the Clipper library to compute the union of the input loops and filter them by the specified minimum area.
- Converts the resulting loops back to Revit curves and ensures the curves are properly chained.
- If `drawCurvesInRevit` is [true](#), the resulting loops are drawn in the Revit document.

If an error occurs during processing, the method returns a failure result with an appropriate error message.

# Namespace MBF.Revit.Data.Constants

## Enums

### [MbCeilingType](#)

Represents the type of ceiling used in a building or structure.

# Enum MbCeilingType

Namespace: [MBF.Revit.Data.Constants](#)

Assembly: MBF.Revit.Data.dll

Represents the type of ceiling used in a building or structure.

```
public enum MbCeilingType
```

## Fields

**Exposed = 3**

Represents the exposed state of an object or entity. This value is typically used to indicate that the object or entity is in a publicly visible or accessible state.

**PlasterBoard = 2**

Represents plasterboard material in the enumeration.

**TiledMajor = 1**

Represents a tiled layout with major elements emphasized. This value is typically used to specify a layout style where major elements are prominently displayed in a tiled arrangement.

## Remarks

This enumeration defines common ceiling types, such as tiled major ceilings, plasterboard ceilings, and exposed ceilings. It can be used to specify or categorize ceiling styles in construction or design contexts.

# Namespace MBF.Revit.Data.Extensions

## Classes

### [DirectionExtensions](#)

Provides extension methods for working with Autodesk.Revit.DB.XYZ vectors.

### [DocumentExtensions](#)

Provides extension methods for working with Revit documents, enabling operations such as retrieving elements, creating views, and managing linked documents.

### [PipeExtensions](#)

Provides extension methods for working with Autodesk.Revit.DB.Plumbing.Pipe objects in Revit.

### [SelectionExtensions](#)

Provides extension methods for retrieving selected elements of specific types from a UIDocument.

### [SpaceExtensions](#)

Provides extension methods for working with Autodesk.Revit.DB.Mechanical.Space objects, including retrieving boundary segments and validating the space.

### [ViewExtensions](#)

Provides extension methods for working with Revit views.

# Class DirectionExtensions

Namespace: [MBF.Revit.Data.Extensions](#)

Assembly: MBF.Revit.Data.dll

Provides extension methods for working with Autodesk.Revit.DB.XYZ vectors.

```
public static class DirectionExtensions
```

## Inheritance

[object](#) ← DirectionExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class contains methods that extend the functionality of Autodesk.Revit.DB.XYZ objects, enabling operations such as calculating perpendicular vectors.

## Methods

### PerpendicularClockwise(XYZ)

Computes a vector that is perpendicular to the given vector in the clockwise direction within the XY plane.

```
public static XYZ PerpendicularClockwise(this XYZ vector2)
```

#### Parameters

**vector2** XYZ

The input vector for which the perpendicular clockwise vector is calculated.

#### Returns

XYZ

A new Autodesk.Revit.DB.XYZ instance representing the perpendicular clockwise vector in the XY plane. If the input vector is effectively zero in the XY plane, the method returns the unit vector in the X direction.

## Remarks

The method assumes the input vector lies in the XY plane and ignores the Z component. If the input vector is nearly zero in both the X and Y components (within a tolerance of 0.001), the method defaults to returning the unit vector in the X direction.

# Class DocumentExtensions

Namespace: [MBF.Revit.Data.Extensions](#)

Assembly: MBF.Revit.Data.dll

Provides extension methods for working with Revit documents, enabling operations such as retrieving elements, creating views, and managing linked documents.

```
public static class DocumentExtensions
```

## Inheritance

[object](#) ← DocumentExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This static class contains a variety of helper methods designed to simplify common tasks when working with Autodesk Revit's API. These methods include retrieving specific types of elements (e.g., family types, levels, rebar shapes), creating and managing views, and handling linked documents.

Each method in this class is implemented as an extension method for the Autodesk.Revit.DB.Document class, allowing them to be called directly on instances of Autodesk.Revit.DB.Document.

The methods in this class return results wrapped in [MbResult<T>](#) objects, which encapsulate both the operation's outcome and any error messages, ensuring robust error handling.

## Methods

### CreateAndPlaceColumnAtPoint(Document, XYZ, FamilySymbol, Level, Level)

Creates and places a column at a specified point in the Revit document.

```
public static MbResult<FamilyInstance> CreateAndPlaceColumnAtPoint(this Document doc, XYZ  
point, FamilySymbol columnType, Level baseLevel, Level topLevel)
```

## Parameters

**doc** Document

The Revit document where the column will be placed.

**point** XYZ

The XYZ point where the column will be placed.

**columnType** FamilySymbol

The FamilySymbol representing the column type.

**baseLevel** Level

The base level of the column.

**topLevel** Level

The top level of the column.

Returns

[MbResult](#)<FamilyInstance>

An [MbResult<T>](#) containing:

- The created FamilyInstance if successful.
- An error message if any input is invalid or column creation fails.

## GetAllRevitDocuments(Document)

Retrieves the base document and all linked Revit documents.

```
public static List<Document> GetAllRevitDocuments(this Document doc)
```

Parameters

**doc** Document

The base Revit document.

Returns

[List](#)<Document>

A list of Revit Autodesk.Revit.DB.Document instances including the base and all linked documents.

## GetFamilyTypesByCategoryOrderedByName(Document, BuiltInCategory)

Retrieves all family types in the specified category from the given Revit document, ordered by their names.

```
public static MbResult<ObservableCollection<FamilySymbol>>
GetFamilyTypesByCategoryOrderedByName(this Document doc, BuiltInCategory category)
```

### Parameters

**doc** Document

The Revit document from which to retrieve the family types. Cannot be [null](#).

**category** BuiltInCategory

The built-in category to filter the family types.

### Returns

[MbResult<ObservableCollection><FamilySymbol>](#)

A result object containing an observable collection of Autodesk.Revit.DB.FamilySymbol objects ordered by name if the operation succeeds, or an error message in bullet points if the operation fails.

### Remarks

This method uses a filtered element collector to retrieve family types of the specified category and orders them alphabetically by their names.

If an error occurs during the operation, the result will contain a failure message with details in bullet points:

- The Revit document cannot be null.
- An error occurred: [error message]

## GetLevelsOrderedByProperty<TKey>(Document, Func<Level, TKey>, bool)

Retrieves all levels from the specified Revit document as an ObservableCollection, ordered dynamically by a specified property.

```
public static MbResult<ObservableCollection<Level>> GetLevelsOrderedByProperty<TKey>(this Document doc, Func<Level, TKey> keySelector, bool ascending = true)
```

## Parameters

**doc** Document

The Revit document to search for levels.

**keySelector** [Func](#)<Level, TKey>

A function to extract a key from a Level for ordering.

**ascending** [bool](#)

If true, orders levels in ascending order; otherwise, descending.

## Returns

[MbResult](#)<ObservableCollection<Level>>

An [MbResult](#)<T> containing:

- An ObservableCollection of Level objects if successful.
- An error message if the document or keySelector is null.

## Type Parameters

**TKey**

The type of the property used for ordering.

## GetLinkedDocumentsOnly(Document)

Retrieves only the linked Revit documents associated with the given base document.

```
public static List<Document> GetLinkedDocumentsOnly(this Document doc)
```

## Parameters

**doc** Document

The base Revit document.

Returns

[List](#) <Document>

A list of linked Autodesk.Revit.DB.Document instances.

## GetRebarBarTypes(Document)

Retrieves all rebar bar types from the specified Revit document as an ObservableCollection.

```
public static MbResult<ObservableCollection<RebarBarType>> GetRebarBarTypes(this Document doc)
```

Parameters

**doc** Document

The Revit document to retrieve rebar bar types from.

Returns

[MbResult](#)<[ObservableCollection](#) <RebarBarType>>

An [MbResult<T>](#) containing:

- An ObservableCollection of RebarBarType objects if successful.
- An error message if the document is null.

## GetRebarShapes(Document)

Retrieves all rebar shapes from the specified Revit document.

```
public static MbResult<List<RebarShape>> GetRebarShapes(this Document doc)
```

Parameters

**doc** Document

The Revit document to retrieve rebar shapes from.

## Returns

[MbResult](#)<[List](#)<RebarShape>>

An [MbResult<T>](#) containing:

- A list of RebarShape objects if successful.
- An error message if the document is null.

## SearchOrCreate3DView(Document, string, bool)

Searches for a 3D view with the specified name. If it exists, returns it. Optionally, creates a new 3D view if specified.

```
public static MbResult<View3D> SearchOrCreate3DView(this Document doc, string viewName, bool  
createView = false)
```

## Parameters

**doc** Document

The Revit document to search or create the 3D view in.

**viewName** [string](#)

The name of the 3D view to search for or create.

**createView** [bool](#)

If [true](#), creates a new 3D view if one does not exist.

## Returns

[MbResult](#)<View3D>

An [MbResult<T>](#) containing:

- The existing or newly created Autodesk.Revit.DB.View3D object if successful.
- An error message if the document is null, the view name is invalid, or creation fails.

# Class PipeExtensions

Namespace: [MBF.Revit.Data.Extensions](#)

Assembly: MBF.Revit.Data.dll

Provides extension methods for working with Autodesk.Revit.DB.Plumbing.Pipe objects in Revit.

```
public static class PipeExtensions
```

## Inheritance

[object](#) ← PipeExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes utility methods for retrieving geometric and system-related information about pipes, such as endpoints, connectors, curves, and associated piping system types. Each method returns an [MbResult<T>](#) to encapsulate the result or an error message, ensuring robust error handling for common issues like null inputs or invalid pipe states.

## Methods

### GetEndPoints(Pipe)

Retrieves the endpoints of a pipe as a list of XYZ points.

```
public static MbResult<List<XYZ>> GetEndPoints(this Pipe pipe)
```

#### Parameters

**pipe** Pipe

The pipe to get the endpoints for.

#### Returns

## [MbResult<List<XYZ>>](#)

An [MbResult<T>](#) containing the list of endpoints or an error message. Possible error messages:

- "The pipe cannot be null." - Returned if the input pipe is null.
- "The pipe does not have a valid LocationCurve." - Returned if the pipe's location is not a valid curve.

## GetNearestEndPoints(Pipe, XYZ)

Finds the nearest and farthest endpoints of a pipe relative to a specified point.

```
public static MbResult<List<XYZ>> GetNearestEndPoints(this Pipe pipe, XYZ point)
```

### Parameters

**pipe** Pipe

The pipe to find endpoints for.

**point** XYZ

The reference point for distance comparison.

### Returns

## [MbResult<List<XYZ>>](#)

An [MbResult<T>](#) containing a list of nearest and farthest endpoints or an error message. Possible error messages:

- "The pipe cannot be null." - Returned if the input pipe is null.
- "The reference point cannot be null." - Returned if the input point is null.
- "The pipe does not have a valid LocationCurve." - Returned if the pipe's location is not a valid curve.

## GetNearestOneSideConnector(Pipe, XYZ)

Finds the nearest one-side connector on a pipe to a specified point.

```
public static MbResult<Connector> GetNearestOneSideConnector(this Pipe pipe, XYZ point)
```

## Parameters

### **pipe** Pipe

The pipe to search for connectors.

### **point** XYZ

The point to measure the distance from.

## Returns

### [MbResult<Connector>](#)

An [MbResult<T>](#) containing the nearest connector or an error message. Possible error messages:

- "The pipe cannot be null." - Returned if the input pipe is null.
- "The reference point cannot be null." - Returned if the input point is null.
- "No connectors found on the pipe." - Returned if the pipe does not have any valid connectors.
- "An error occurred while finding the nearest connector: {ErrorMessage}" - Returned if an unexpected exception occurs.

## GetPipeCurve(Pipe)

Gets the curve representing the pipe's geometry.

```
public static MbResult<Curve> GetPipeCurve(this Pipe pipe)
```

## Parameters

### **pipe** Pipe

The pipe to retrieve the curve from.

## Returns

### [MbResult<Curve>](#)

An [MbResult<T>](#) containing the curve or an error message. Possible error messages:

- "The pipe cannot be null." - Returned if the input pipe is null.
- "The pipe does not have a valid LocationCurve." - Returned if the pipe's location is not a valid curve.

# GetPipingSystemType(Pipe, Document)

Gets the piping system type associated with a pipe.

```
public static MbResult<PipingSystemType> GetPipingSystemType(this Pipe pipe,  
Document document)
```

## Parameters

**pipe** Pipe

The pipe to retrieve the piping system type for.

**document** Document

The Revit document containing the pipe.

## Returns

[MbResult<PipingSystemType>](#)

An [MbResult<T>](#) containing the piping system type or an error message. Possible error messages:

- "The pipe cannot be null." - Returned if the input pipe is null.
- "The document cannot be null." - Returned if the input document is null.
- "The pipe is not associated with a piping system." - Returned if the pipe is not part of a piping system.
- "Failed to retrieve the piping system type." - Returned if the piping system type cannot be retrieved.
- "An error occurred while retrieving the piping system type: {ErrorMessage}" - Returned if an unexpected exception occurs.

# Class SelectionExtensions

Namespace: [MBF.Revit.Data.Extensions](#)

Assembly: MBF.Revit.Data.dll

Provides extension methods for retrieving selected elements of specific types from a UIDocument.

```
public static class SelectionExtensions
```

## Inheritance

[object](#) ← SelectionExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### GetSelectedElements<T>(UIDocument)

Gets the selected elements of the specified type from the document.

```
public static List<T> GetSelectedElements<T>(this UIDocument uiDocument) where T : Element
```

#### Parameters

**uiDocument** UIDocument

The UIDocument to get the selected elements from.

#### Returns

[List](#)<T>

A list of the selected elements of the specified type.

#### Type Parameters

T

The type of elements to get.

## Remarks

Error messages:

- Returns an empty list if no elements of the specified type are selected.
- Throws ArgumentNullException if **uiDocument** is null.

## GetSelectedFamilyInstances(UIDocument)

Gets the selected family instances from the document.

```
public static List<FamilyInstance> GetSelectedFamilyInstances(this UIDocument uiDocument)
```

## Parameters

**uiDocument** UIDocument

The UI document to get the selected family instances from.

## Returns

[List](#) <FamilyInstance>

A list of the selected family instances.

## Remarks

Error messages:

- Returns an empty list if no family instances are selected.

## GetSelectedPipes(UIDocument)

Gets the selected pipes from the document.

```
public static List<Pipe> GetSelectedPipes(this UIDocument uiDocument)
```

## Parameters

## **uiDocument** UIDocument

The UI document to get the selected pipes from.

### Returns

[List](#) <Pipe>

A list of the selected pipes.

### Remarks

Error messages:

- Returns an empty list if no pipes are selected.

## GetSelectedSpaces(UIDocument)

Gets the selected spaces from the document.

```
public static List<Space> GetSelectedSpaces(this UIDocument uiDocument)
```

### Parameters

#### **uiDocument** UIDocument

The UI document to get the selected spaces from.

### Returns

[List](#) <Space>

A list of the selected spaces.

### Remarks

Error messages:

- Returns an empty list if no spaces are selected.

## GetSelectedSprinklers(UIDocument)

Gets the selected sprinklers from the document. Filters the selected FamilyInstance elements to return only those categorized as BuiltInCategory.OST\_Sprinklers.

```
public static List<FamilyInstance> GetSelectedSprinklers(this UIDocument uiDocument)
```

## Parameters

**uiDocument** UIDocument

The UI document to get the selected sprinklers from.

## Returns

[List](#)<FamilyInstance>

A list of the selected sprinkler family instances.

## Remarks

Error messages:

- Returns an empty list if no sprinklers are selected.

# Class SpaceExtensions

Namespace: [MBF.Revit.Data.Extensions](#)

Assembly: MBF.Revit.Data.dll

Provides extension methods for working with Autodesk.Revit.DB.Mechanical.Space objects, including retrieving boundary segments and validating the space.

```
public static class SpaceExtensions
```

## Inheritance

[object](#) ← SpaceExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class contains methods to assist in common operations related to Autodesk.Revit.DB.Mechanical.Space objects, such as extracting boundary curves and ensuring the validity of a space. These methods are designed to simplify working with spaces in spatial modeling scenarios.

## Methods

### GetBoundarySegment(Space)

Retrieves the boundary segments of a space as a list of curves.

```
public static MbResult<List<Curve>> GetBoundarySegment(this Space space)
```

#### Parameters

**space** Space

The space to get the boundary segments for.

#### Returns

## [MbResult](#)<[List](#)><Curve>>

An [MbResult<T>](#) containing:

- A list of boundary curves if the operation is successful.
- An error message if the operation fails. Possible error messages include:

## Validate(Space)

Validates the space object to ensure it is valid and has a positive area.

```
public static MbResult<bool> Validate(this Space space)
```

### Parameters

**space** Space

The space to validate.

### Returns

[MbResult](#)<[bool](#)>

An [MbResult<T>](#) containing:

- **true** if the space is valid.
- An error message if the validation fails. Possible error messages include:

# Class ViewExtensions

Namespace: [MBF.Revit.Data.Extensions](#)

Assembly: MBF.Revit.Data.dll

Provides extension methods for working with Revit views.

```
public static class ViewExtensions
```

## Inheritance

[object](#) ← ViewExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class contains methods that extend the functionality of Revit views, enabling operations such as retrieving visible elements.

## Methods

### GetVisibleElements(View, Document)

Retrieves all elements that are visible in the specified Revit view.

```
public static MbResult<List<Element>> GetVisibleElements(this View view, Document document)
```

#### Parameters

**view** View

The Revit view to search for visible elements.

**document** Document

The Revit document containing the view.

## Returns

[MbResult<List<Element>>](#)

An [MbResult<T>](#) object containing:

- A list of elements visible in the specified view if the operation is successful.
- A failure result with the following error messages for invalid parameters:
  - "The view parameter cannot be null."
  - "The document parameter cannot be null."

# Namespace MBF.Revit.Data.Models

## Classes

### [BeamProcessingResult](#)

Result of beam processing operations

### [BoundaryLines](#)

Represents boundary lines for the grid system

### [CeilingAnalysisResult](#)

Result of ceiling analysis operations

### [CeilingData](#)

Data about a single ceiling element

### [CeilingFaceGrid](#)

Represents a grid system for placing sprinklers on ceiling faces

### [CeilingPattern](#)

Represents the pattern extracted from a ceiling

### [CeilingPatternExtractor](#)

Extracts ceiling pattern information

### [GridLine](#)

Represents a line in the grid system with its type classification

### [GridLineGenerator](#)

Generates grid lines for the ceiling grid system

### [GridPointInfo](#)

Represents information about a grid point

### [GridValidationResult](#)

Represents the result of grid validation

### [MbDocument](#)

Represents a Revit document, including both standalone documents and linked documents.

# Class BeamProcessingResult

Namespace: [MBF.Revit.Data.Models](#)

Assembly: MBF.Revit.Data.dll

Result of beam processing operations

```
public class BeamProcessingResult
```

## Inheritance

[object](#) ← BeamProcessingResult

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### BeamProcessingResult()

```
public BeamProcessingResult()
```

## Properties

### BeamSortingPoint

```
public XYZ BeamSortingPoint { get; set; }
```

Property Value

XYZ

### CrossBeams

```
public List<Tuple<Line, double, double>> CrossBeams { get; set; }
```

Property Value

[List](#) <[Tuple](#) <Line, [double](#), [double](#)>>

MainBeams

```
public List<Tuple<Line, double, double>> MainBeams { get; set; }
```

Property Value

[List](#) <[Tuple](#) <Line, [double](#), [double](#)>>

# Class BoundaryLines

Namespace: [MBF.Revit.Data.Models](#)

Assembly: MBF.Revit.Data.dll

Represents boundary lines for the grid system

```
public class BoundaryLines
```

## Inheritance

[object](#) ← BoundaryLines

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Constructors

## BoundaryLines(Line, Line, Line, Line)

```
public BoundaryLines(Line maxBoundary, Line minBoundary, Line maxSecondaryBoundary,  
Line minSecondaryBoundary)
```

## Parameters

**maxBoundary** Line

**minBoundary** Line

**maxSecondaryBoundary** Line

**minSecondaryBoundary** Line

# Properties

## MaxBoundary

```
public Line MaxBoundary { get; }
```

Property Value

Line

## MaxSecondaryBoundary

```
public Line MaxSecondaryBoundary { get; }
```

Property Value

Line

## MinBoundary

```
public Line MinBoundary { get; }
```

Property Value

Line

## MinSecondaryBoundary

```
public Line MinSecondaryBoundary { get; }
```

Property Value

Line

# Class CeilingAnalysisResult

Namespace: [MBF.Revit.Data.Models](#)

Assembly: MBF.Revit.Data.dll

Result of ceiling analysis operations

```
public class CeilingAnalysisResult
```

## Inheritance

[object](#) ← CeilingAnalysisResult

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### CeilingAnalysisResult()

```
public CeilingAnalysisResult()
```

## Properties

### CeilingType

```
public MbCeilingType CeilingType { get; set; }
```

Property Value

[MbCeilingType](#)

### ElevationPoint

```
public XYZ ElevationPoint { get; set; }
```

Property Value

XYZ

## MajorCeiling

```
public Ceiling MajorCeiling { get; set; }
```

Property Value

Ceiling

## MajorFace

```
public Face MajorFace { get; set; }
```

Property Value

Face

# Class CeilingData

Namespace: [MBF.Revit.Data.Models](#)

Assembly: MBF.Revit.Data.dll

Data about a single ceiling element

```
public class CeilingData
```

## Inheritance

[object](#) ← CeilingData

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Constructors

## CeilingData()

```
public CeilingData()
```

# Properties

## Ceiling

```
public Ceiling Ceiling { get; set; }
```

## Property Value

Ceiling

## Face

```
public Face Face { get; set; }
```

Property Value

Face

Type

```
public MbCeilingType Type { get; set; }
```

Property Value

[MbCeilingType](#)

# Class CeilingFaceGrid

Namespace: [MBF.Revit.Data.Models](#)

Assembly: MBF.Revit.Data.dll

Represents a grid system for placing sprinklers on ceiling faces

```
public class CeilingFaceGrid
```

## Inheritance

[object](#) ← CeilingFaceGrid

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Properties

## MaxDimensionLines

Gets a read-only collection of grid lines representing the maximum dimensions of the grid.

```
public IReadOnlyList<GridLine> MaxDimensionLines { get; }
```

## Property Value

[IReadOnlyList](#) <[GridLine](#)>

## Remarks

The returned collection is immutable and reflects the maximum dimension lines as defined by the grid's configuration.

## MinDimensionLines

Gets a read-only list of grid lines representing the minimum dimensions of the grid.

```
public IReadOnlyList<GridLine> MinDimensionLines { get; }
```

## Property Value

[IReadOnlyList](#) <GridLine>

## Methods

**Create(Ceiling, Document, RevitLinkInstance, Space, CeilingFaceGridParameters, double, List<XYZ>, double)**

Creates a new instance of [CeilingFaceGrid](#) based on the specified ceiling, space, and parameters.

```
public static MbResult<CeilingFaceGrid> Create(Ceiling ceiling, Document document,
RevitLinkInstance architecturalLink, Space space, CeilingFaceGridParameters parameters,
double ceilingHeight, List<XYZ> spacePolygonCorners, double regularOffsetFeet)
```

## Parameters

**ceiling** Ceiling

The ceiling element for which the grid is being created. Cannot be null.

**document** Document

The Revit document containing the ceiling and related elements. Cannot be null.

**architecturalLink** RevitLinkInstance

The Revit link instance representing the architectural model. Cannot be null.

**space** Space

The space in which the ceiling resides. Cannot be null.

**parameters** [CeilingFaceGridParameters](#)

The parameters that define the grid's configuration. Cannot be null.

**ceilingHeight** [double](#)

The height of the ceiling in feet. Must be a positive value.

**spacePolygonCorners** [List](#) <XYZ>

The list of corner points defining the polygonal boundary of the space. Cannot be null or empty.

## `regularOffsetFeet` [double](#)

The regular offset, in feet, used for grid generation. Must be a positive value.

## Returns

### [MbResult](#)<[CeilingFaceGrid](#)>

An [MbResult](#)<T> containing the created [CeilingFaceGrid](#) if the operation succeeds; otherwise, an error message describing the failure.

## Remarks

This method validates the input parameters, configures the grid based on the ceiling and space, and generates the grid using the specified offset. If any step in the process fails, the method returns a failure result with an appropriate error message.

## `GetGridPoint(int, int)`

Retrieves a grid point from the collection based on the specified indices.

```
public MbResult<XYZ> GetGridPoint(int maxIndex, int minIndex)
```

## Parameters

### `maxIndex` [int](#)

The maximum index of the grid point to retrieve. Must be a valid index within the grid.

### `minIndex` [int](#)

The minimum index of the grid point to retrieve. Must be a valid index within the grid.

## Returns

### [MbResult](#)<XYZ>

A [MbResult](#)<T> containing the grid point as an Autodesk.Revit.DB.XYZ object if the operation is successful; otherwise, a failure result with an error message.

## Remarks

This method validates the indices and checks whether the grid point is initialized and valid. If the indices are invalid or the grid point is not initialized, the method returns a failure result.

## GetGridPointDescription(int, int)

Retrieves the description of a grid point based on the specified indices.

```
public MbResult<string> GetGridPointDescription(int maxIndex, int minIndex)
```

### Parameters

**maxIndex** [int](#)

The maximum index of the grid point. Must be a valid index within the grid's bounds.

**minIndex** [int](#)

The minimum index of the grid point. Must be a valid index within the grid's bounds.

### Returns

[MbResult<string>](#)

An [MbResult<T>](#) containing the description of the grid point if the indices are valid and the grid is initialized; otherwise, a failure result with an appropriate error message.

### Remarks

If the grid point description is null, an empty string is returned as the description.

## GetValidGridPoints()

Retrieves a list of valid grid points from the current grid.

```
public MbResult<List<GridPointInfo>> GetValidGridPoints()
```

### Returns

[MbResult<List<GridPointInfo>>](#)

A [MbResult<T>](#) containing a list of [GridPointInfo](#) objects representing the valid grid points. If the grid is not initialized, the result is a failure with an appropriate error message.

## Remarks

A grid point is considered valid if its Z-coordinate does not match the invalid Z-coordinate value specified in the grid parameters. If the grid is not initialized, the method returns a failure result.

## ValidateGrid()

Validates the current grid and returns a detailed validation result.

```
public MbResult<GridValidationResult> ValidateGrid()
```

## Returns

[MbResult<GridValidationResult>](#)

An [MbResult<T>](#) containing a [GridValidationResult](#) object that provides details about the validation, including the total number of points, valid points, invalid points, the percentage of valid points, and any error or warning messages.

## Remarks

This method evaluates the grid's validity based on its initialization state, the number of valid grid points, and the percentage of valid points relative to the total points. If the grid is not initialized or contains no valid points, the validation will fail. A warning is included if the percentage of valid points falls below a predefined threshold.

# Class CeilingPattern

Namespace: [MBF.Revit.Data.Models](#)

Assembly: MBF.Revit.Data.dll

Represents the pattern extracted from a ceiling

```
public class CeilingPattern
```

## Inheritance

[object](#) ← CeilingPattern

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

CeilingPattern(Line, double, Line, double, Face)

```
public CeilingPattern(Line primaryLine, double primarySpacing, Line secondaryLine, double  
secondarySpacing, Face face)
```

## Parameters

**primaryLine** Line

**primarySpacing** [double](#)

**secondaryLine** Line

**secondarySpacing** [double](#)

**face** Face

## Properties

## Face

```
public Face Face { get; }
```

### Property Value

Face

## PrimaryLine

```
public Line PrimaryLine { get; }
```

### Property Value

Line

## PrimarySpacing

```
public double PrimarySpacing { get; }
```

### Property Value

[double](#)

## SecondaryLine

```
public Line SecondaryLine { get; }
```

### Property Value

Line

## SecondarySpacing

```
public double SecondarySpacing { get; }
```

Property Value

[double](#) ↗

# Class CeilingPatternExtractor

Namespace: [MBF.Revit.Data.Models](#)

Assembly: MBF.Revit.Data.dll

Extracts ceiling pattern information

```
public static class CeilingPatternExtractor
```

## Inheritance

[object](#) ← CeilingPatternExtractor

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### GetCeilingPattern(Ceiling, Document, RevitLinkInstance)

```
public static MbResult<CeilingPattern> GetCeilingPattern(Ceiling ceiling, Document  
parentDoc, RevitLinkInstance rli)
```

#### Parameters

**ceiling** Ceiling

**parentDoc** Document

**rli** RevitLinkInstance

#### Returns

[MbResult<CeilingPattern>](#)

# Class GridLine

Namespace: [MBF.Revit.Data.Models](#)

Assembly: MBF.Revit.Data.dll

Represents a line in the grid system with its type classification

```
public class GridLine
```

## Inheritance

[object](#) ← GridLine

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Constructors

## GridLine(string, Line)

```
public GridLine(string type, Line line)
```

## Parameters

**type** [string](#)

**line** Line

# Properties

## Line

```
public Line Line { get; }
```

## Property Value

Line

## Type

```
public string Type { get; }
```

Property Value

[string](#) ↗

# Class GridLineGenerator

Namespace: [MBF.Revit.Data.Models](#)

Assembly: MBF.Revit.Data.dll

Generates grid lines for the ceiling grid system

```
public static class GridLineGenerator
```

## Inheritance

[object](#) ← GridLineGenerator

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### GenerateDimensionLines(Line, double, XYZ, List<Line>)

```
public static MbResult<List<GridLine>> GenerateDimensionLines(Line baseLine, double spacing,  
XYZ faceCenter, List<Line> spacePolygonLines)
```

## Parameters

**baseLine** Line

**spacing** [double](#)

**faceCenter** XYZ

**spacePolygonLines** [List](#)<Line>

## Returns

[MbResult](#)<[List](#)<[GridLine](#)>>

# Class GridPointInfo

Namespace: [MBF.Revit.Data.Models](#)

Assembly: MBF.Revit.Data.dll

Represents information about a grid point

```
public class GridPointInfo
```

## Inheritance

[object](#) ← GridPointInfo

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

GridPointInfo(int, int, XYZ, string)

```
public GridPointInfo(int maxIndex, int minIndex, XYZ point, string description)
```

## Parameters

maxIndex [int](#)

minIndex [int](#)

point XYZ

description [string](#)

## Properties

### Description

```
public string Description { get; }
```

Property Value

[string](#) ↗

## MaxIndex

```
public int MaxIndex { get; }
```

Property Value

[int](#) ↗

## MinIndex

```
public int MinIndex { get; }
```

Property Value

[int](#) ↗

## Point

```
public XYZ Point { get; }
```

Property Value

XYZ

# Class GridValidationResult

Namespace: [MBF.Revit.Data.Models](#)

Assembly: MBF.Revit.Data.dll

Represents the result of grid validation

```
public class GridValidationResult
```

## Inheritance

[object](#) ← GridValidationResult

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### GridValidationResult()

```
public GridValidationResult()
```

## Properties

### ErrorMessages

```
public List<string> ErrorMessages { get; }
```

Property Value

[List](#)<[string](#)>

### InvalidPoints

```
public int InvalidPoints { get; set; }
```

Property Value

[int ↗](#)

IsValid

```
public bool IsValid { get; set; }
```

Property Value

[bool ↗](#)

TotalPoints

```
public int TotalPoints { get; set; }
```

Property Value

[int ↗](#)

ValidPoints

```
public int ValidPoints { get; set; }
```

Property Value

[int ↗](#)

ValidPointsPercentage

```
public double ValidPointsPercentage { get; set; }
```

Property Value

[double](#)

## WarningMessages

```
public List<string> WarningMessages { get; }
```

Property Value

[List](#) <[string](#)>

# Class MbDocument

Namespace: [MBF.Revit.Data.Models](#)

Assembly: MBF.Revit.Data.dll

Represents a Revit document, including both standalone documents and linked documents.

```
public class MbDocument
```

## Inheritance

[object](#) ← MbDocument

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

The [MbDocument](#) class provides functionality to manage and retrieve information about Revit documents, including linked documents. It supports operations such as retrieving all linked documents associated with a base document and accessing properties like the transformation applied to a linked document or its parent document.

## Constructors

### MbDocument(Document)

Initializes a new instance of the [MbDocument](#) class for a standalone (non-linked) document.

```
public MbDocument(Document document)
```

## Parameters

**document** Document

The standalone Revit document.

## MbDocument(RevitLinkInstance, Document)

Initializes a new instance of the [MbDocument](#) class for a linked document.

```
public MbDocument(RevitLinkInstance linkInstance, Document parentDocument)
```

### Parameters

**linkInstance** RevitLinkInstance

The Revit link instance referencing the linked document.

**parentDocument** Document

The parent document containing the link.

## Properties

### Document

Gets or sets the associated Revit [Document](#).

```
public Document Document { get; set; }
```

### Property Value

Document

### IsLinked

Gets or sets a value indicating whether this document is a linked document.

```
public bool IsLinked { get; set; }
```

### Property Value

[bool](#)

### LinkInstance

Gets or sets the Autodesk.Revit.DB.RevitLinkInstance that defines the link context.

```
public RevitLinkInstance LinkInstance { get; set; }
```

Property Value

RevitLinkInstance

## LinkParentDocument

Gets or sets the parent document from which this link was referenced, if applicable.

```
public Document LinkParentDocument { get; set; }
```

Property Value

Document

## LinkTransform

Gets or sets the transformation applied to the link instance.

```
public Transform LinkTransform { get; set; }
```

Property Value

Transform

## Methods

### GetAllDocuments(Document)

Retrieves a list of [MbDocument](#) instances including the base document and all linked documents.

```
public static List<MbDocument> GetAllDocuments(Document doc)
```

Parameters

**doc** Document

The base Revit document.

Returns

[List](#) <[MbDocument](#)>

A list of [MbDocument](#) objects.

## GetLinkedDocuments(Document)

Retrieves all linked documents associated with the given base document.

```
public static List<MbDocument> GetLinkedDocuments(Document doc)
```

Parameters

**doc** Document

The base Revit document.

Returns

[List](#) <[MbDocument](#)>

A list of [MbDocument](#) objects representing the linked documents.

# Namespace MBF.Revit.Data.SelectionFilters

## Classes

### [DynamicCategorySelectionFilter](#)

Represents a selection filter that dynamically allows elements based on their categories.

### [FaceSelectionFilter](#)

Provides a selection filter for Revit elements and references, allowing selection of elements based on their category and ensuring that references correspond to geometric faces.

### [HostOrLinkedCategorySelectionFilter](#)

A selection filter that allows selecting elements from the host or a linked Revit model, restricted to a specific set of categories.

### [LinkedCategorySelectionFilter](#)

Selection filter for linked Revit documents that restricts selection to elements belonging to specified categories within a selected MBF.Revit.Data.SelectionFilters.LinkedCategorySelectionFilter.RevitLink Instance.

# Class DynamicCategorySelectionFilter

Namespace: [MBF.Revit.Data.SelectionFilters](#)

Assembly: MBF.Revit.Data.dll

Represents a selection filter that dynamically allows elements based on their categories.

```
public class DynamicCategorySelectionFilter : ISelectionFilter
```

## Inheritance

[object](#) ← DynamicCategorySelectionFilter

## Implements

ISelectionFilter

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This filter is designed to restrict selection to elements belonging to specific categories. It supports both Revit 2024+ (long-based ElementId) and earlier versions (int-based ElementId).

## Constructors

### DynamicCategorySelectionFilter(params BuiltInCategory[])

Initializes a new instance of the [DynamicCategorySelectionFilter](#) class with the specified categories that should be allowed during selection.

```
public DynamicCategorySelectionFilter(params BuiltInCategory[] categories)
```

## Parameters

**categories** BuiltInCategory[]

An array of BuiltInCategory values to allow.

# Methods

## AllowElement(Element)

Determines whether a specific element is allowed for selection.

```
public bool AllowElement(Element element)
```

Parameters

**element** Element

The element being evaluated.

Returns

bool ↗

**true** if the element belongs to one of the allowed categories; otherwise, **false**.

## AllowReference(Reference, XYZ)

Determines whether a reference (face, edge, etc.) is allowed for selection. This implementation disallows reference-based selection and only allows full element selection.

```
public bool AllowReference(Reference reference, XYZ position)
```

Parameters

**reference** Reference

The reference being evaluated.

**position** XYZ

The 3D position associated with the reference.

Returns

bool ↗

Always returns **false** to disable reference selection.



# Class FaceSelectionFilter

Namespace: [MBF.Revit.Data.SelectionFilters](#)

Assembly: MBF.Revit.Data.dll

Provides a selection filter for Revit elements and references, allowing selection of elements based on their category and ensuring that references correspond to geometric faces.

```
public class FaceSelectionFilter : ISelectionFilter
```

## Inheritance

[object](#) ← FaceSelectionFilter

## Implements

ISelectionFilter

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This filter can be used to restrict element selection to specific categories or to ensure that only face references are considered valid. It supports two modes of operation:

- Allowing all element categories by default.
- Restricting selection to specific categories by providing a list of allowed categories.

The filter is particularly useful in scenarios where precise control over element or reference selection is required, such as in custom Revit add-ins or tools.

## Constructors

### FaceSelectionFilter(Document)

Initializes a new instance of the [FaceSelectionFilter](#) that allows all element categories.

```
public FaceSelectionFilter(Document document)
```

## Parameters

## **document** Document

The current Revit document.

## FaceSelectionFilter(Document, BuiltInCategory)

Initializes a new instance of the [FaceSelectionFilter](#) that only allows elements from the specified category.

```
public FaceSelectionFilter(Document document, BuiltInCategory category)
```

### Parameters

#### **document** Document

The current Revit document.

#### **category** BuiltInCategory

A single allowed BuiltInCategory for selection.

## Methods

### AllowElement(Element)

Determines whether the given element is eligible for selection based on its category.

```
public bool AllowElement(Element element)
```

### Parameters

#### **element** Element

The element to evaluate.

### Returns

#### [bool](#)

**true** if the element is in the allowed category list (or all are allowed); otherwise, **false**.

## AllowReference(Reference, XYZ)

Determines whether the referenced geometry is a face.

```
public bool AllowReference(Reference refer, XYZ point)
```

### Parameters

**refer** Reference

The reference to check.

**point** XYZ

The point on the geometry (not used).

### Returns

bool ↗

**true** if the reference is to a face; otherwise, **false**.

# Class HostOrLinkedCategorySelectionFilter

Namespace: [MBF.Revit.Data.SelectionFilters](#)

Assembly: MBF.Revit.Data.dll

A selection filter that allows selecting elements from the host or a linked Revit model, restricted to a specific set of categories.

```
public class HostOrLinkedCategorySelectionFilter : ISelectionFilter
```

## Inheritance

[object](#) ← HostOrLinkedCategorySelectionFilter

## Implements

ISelectionFilter

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

Error messages:

- If `categories` is `null` in the constructor, an empty list is used.
- If `_revitLinkInstance` is `null` in [AllowReference\(Reference, XYZ\)](#), returns `false`.
- If `linkedDoc` is `null` in [AllowReference\(Reference, XYZ\)](#), returns `false`.

## Constructors

### HostOrLinkedCategorySelectionFilter(IEnumerable<Category>)

Initializes a new instance of the [HostOrLinkedCategorySelectionFilter](#) class.

```
public HostOrLinkedCategorySelectionFilter(IEnumerable<Category> categories)
```

## Parameters

`categories` [IEnumerable](#)<Category>

The categories to allow from both host and linked documents.

## Remarks

- If `categories` is `null`, an empty list is used.

# Methods

## AllowElement(Element)

Determines whether an element can be selected directly.

```
public bool AllowElement(Element elem)
```

### Parameters

`elem` Element

The element being evaluated.

### Returns

`bool` ↗

`true` if it's a host element in the allowed categories or a Autodesk.Revit.DB.RevitLinkInstance; otherwise, `false`.

## AllowReference(Reference, XYZ)

Determines whether a reference (e.g. in a linked file) is selectable.

```
public bool AllowReference(Reference reference, XYZ position)
```

### Parameters

`reference` Reference

The reference to the element.

`position` XYZ

The selection point (not used).

Returns

[bool](#) ↗

`true` if the referenced element's category is allowed; otherwise, `false`.

Remarks

- Returns `false` if `_revitLinkInstance` is `null`.
- Returns `false` if `linkedDoc` is `null`.

# Class LinkedCategorySelectionFilter

Namespace: [MBF.Revit.Data.SelectionFilters](#)

Assembly: MBF.Revit.Data.dll

Selection filter for linked Revit documents that restricts selection to elements belonging to specified categories within a selected MBF.Revit.Data.SelectionFilters.LinkedCategorySelectionFilter.RevitLink Instance.

```
public class LinkedCategorySelectionFilter : ISelectionFilter
```

## Inheritance

[object](#) ← LinkedCategorySelectionFilter

## Implements

ISelectionFilter

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### LinkedCategorySelectionFilter(IEnumerable<Category>)

Initializes a new instance of the [LinkedCategorySelectionFilter](#) class.

```
public LinkedCategorySelectionFilter(IEnumerable<Category> categories)
```

## Parameters

**categories** [IEnumerable](#)<Category>

The categories to allow from the linked document.

## Remarks

- If **categories** is **null**, an empty list is used.

# Methods

## AllowElement(Element)

Determines whether an element is selectable. Only allows selecting MBF.Revit.Data.SelectionFilters.LinkedCategorySelectionFilter.RevitLinkInstance elements.

```
public bool AllowElement(Element elem)
```

### Parameters

**elem** Element

The element to evaluate.

### Returns

bool ↗

**true** if the element is a MBF.Revit.Data.SelectionFilters.LinkedCategorySelectionFilter.RevitLinkInstance; otherwise, **false**.

### Remarks

- Returns **false** if the element is not a MBF.Revit.Data.SelectionFilters.LinkedCategorySelectionFilter.RevitLinkInstance.

## AllowReference(Reference, XYZ)

Determines whether a referenced element inside the selected linked model is allowed.

```
public bool AllowReference(Reference reference, XYZ position)
```

### Parameters

**reference** Reference

The reference to an element in the linked model.

**position** XYZ

The point on the element being selected (not used).

## Returns

[bool](#) ↗

**true** if the linked element belongs to one of the allowed categories; otherwise, **false**.

## Remarks

- Returns **false** if no MBF.Revit.Data.SelectionFilters.LinkedCategorySelectionFilter.RevitLinkInstance has been selected.
- Returns **false** if the referenced element does not exist or its category is not allowed.

# Namespace MBF.Revit.Data.Utils

## Classes

### [CeilingPatternDeducer](#)

Provides functionality to deduce the ceiling grid pattern for a given space based on its area, boundary points, and a primary direction.

### [EntityUtils](#)

Provides utility methods for working with entities and their associated fields, enabling the retrieval and assignment of values such as Autodesk.Revit.DB.XYZ and ElementId.

### [GeometryUtils](#)

Utility methods for geometry operations

### [MbCeilingUtils](#)

Provides utility methods for working with Revit ceiling elements.

### [MbConnectorUtilities](#)

Provides utility methods for managing and interacting with MEP connectors in a Revit document.

### [MbCurveUtils](#)

Utility methods for working with curves in Revit.

### [MbFaceUtils](#)

Retrieves the fill pattern associated with the surface material of the specified face.

### [MbFamilyInstanceUtils](#)

Provides utility methods for working with family instances in a Revit model.

### [MbFlexPipesUtils](#)

Filters the provided collection of Revit elements and retrieves all flex pipes.

### [MbLinkedElementUtils](#)

Provides utility methods for working with linked elements in Revit documents, including coordinate transformations between parent and linked files.

### [MbParameterUtils](#)

Provides utility methods for working with Revit parameters.

### [MbPipeAccessoriesUtils](#)

Provides utility methods for working with pipe accessory elements in Revit.

### [MbPipeFittingsUtils](#)

Provides utility methods for working with pipe fittings in Revit.

## [MbPipesUtils](#)

Provides utility methods for working with pipe elements in Revit.

## [MbProjectLocationUtils](#)

Provides utility methods for working with project location transformations in Revit.

## [MbRevitSpaceOldRepoUtils](#)

Utility class for space operations and calculations using only Revit Space objects

## [MbSelectionUtils](#)

Provides utility methods for working with family instances in Revit documents.

## [MbSpaceUtils](#)

Provides utility methods for working with Autodesk.Revit.DB.Mechanical.Space elements in Autodesk Revit.

## [MbSprinklersUtils](#)

Provides utility methods for working with sprinkler elements in Revit.

## [MbSurveyPointUtils](#)

Provides utility methods for working with survey points in Revit documents.

## [MbUnitUtils](#)

Provides utility methods for working with measurement units in Revit.

## [MbViewUtils](#)

Provides utility methods for working with Revit views, including graphical view checks, zooming to elements, creating temporary 3D views, and managing view isolation.

## [RevitUnitUtils](#)

Determines whether the units of the specified Revit document are metric.

## [XyzExtensibleStorage](#)

Provides utilities for reading and writing Autodesk.Revit.DB.XYZ start point data for Autodesk.Revit.DB.Plumbing.Pipe elements using Revit's Extensible Storage API.

# Enums

## [PipePointType](#)

Represents a specific point along a pipe, either at the start or the end.

# Class CeilingPatternDeducer

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides functionality to deduce the ceiling grid pattern for a given space based on its area, boundary points, and a primary direction.

```
public static class CeilingPatternDeducer
```

## Inheritance

[object](#) ← CeilingPatternDeducer

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class contains methods to calculate the main and perpendicular ceiling grid lines, along with their respective offsets, based on the provided space dimensions and configuration parameters. It is designed to handle both small and large spaces by applying different offset values depending on the area threshold.

## Methods

### DeduceCeilingPatternForSpace(double, List<XYZ>, XYZ, double, double, double)

Deduces the ceiling grid pattern for a given space using its area, boundary points, and a primary direction.

```
public static MbResult<(Line maxLine, double maxOffset, Line minLine, double  
minOffset)> DeduceCeilingPatternForSpace(double currentSpaceArea, List<XYZ> spacePoints,  
XYZ primaryDirection, double smallSpaceThreshold, double smallSpaceOffsetFeet,  
double largeSpaceOffsetFeet)
```

## Parameters

`currentSpaceArea` [double](#)

The area of the space in square feet.

`spacePoints` [List](#)<XYZ>

The list of points representing the boundary of the space. Must not be null or empty.

`primaryDirection` XYZ

The primary direction vector for the ceiling grid. Must not be null.

`smallSpaceThreshold` [double](#)

The area threshold (in square feet) below which the space is considered small.

`smallSpaceOffsetFeet` [double](#)

The offset (in feet) to use for small spaces.

`largeSpaceOffsetFeet` [double](#)

The offset (in feet) to use for large spaces.

Returns

[MbResult](#)<(Line [maxLine](#), [double](#) [maxOffset](#), Line [minLine](#), [double](#) [minOffset](#))>

An [MbResult<T>](#) containing a tuple with:

- `maxLine`: The main ceiling grid line in the primary direction.
- `maxOffset`: The offset for the main grid line.
- `minLine`: The perpendicular ceiling grid line.
- `minOffset`: The offset for the perpendicular grid line.

### Possible error messages:

- "Space points cannot be null or empty" - Returned if `spacePoints` is null or contains no elements.
- "Primary direction cannot be null" - Returned if `primaryDirection` is null.
- "Error creating line with provided direction: {ex.Message}" - Returned if an exception occurs while creating the main grid line.
- "Could not determine max dimension line" - Returned if the main grid line could not be created.

# Class EntityUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with entities and their associated fields, enabling the retrieval and assignment of values such as Autodesk.Revit.DB.XYZ and ElementId.

```
public static class EntityUtils
```

## Inheritance

[object](#) ← EntityUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

The [EntityUtils](#) class adapts to different runtime environments to ensure compatibility with varying unit systems. It includes methods for retrieving and setting field values in entities, handling both Autodesk.Revit.DB.XYZ and ElementId types. This class is designed for use in scenarios where precise handling of units and field data is required.

### Error messages:

- Entity or field cannot be null.
- Value cannot be null.
- Failed to retrieve or set the specified field value.

## Methods

### GetElementId(Entity, Field)

Retrieves the ElementId value associated with the specified [field](#) in the given [entity](#).

```
public static ElementId GetElementId(Entity entity, Field field)
```

## Parameters

### **entity** Entity

The Autodesk.Revit.DB.ExtensibleStorage.Entity from which to retrieve the ElementId value. Cannot be [null](#).

### **field** Field

The Autodesk.Revit.DB.ExtensibleStorage.Field that identifies the data to retrieve. Cannot be [null](#).

## Returns

### ElementId

The ElementId value associated with the specified **field** in the **entity**.

## Remarks

The method retrieves the ElementId value using the appropriate unit system based on the runtime environment.

### **Error messages:**

- Entity or field cannot be null.
- Failed to retrieve the ElementId value from the entity.

## GetXYZ(Entity, Field)

Retrieves the Autodesk.Revit.DB.XYZ value of the specified field from the given entity.

```
public static XYZ GetXYZ(Entity entity, Field field)
```

## Parameters

### **entity** Entity

The entity from which to retrieve the Autodesk.Revit.DB.XYZ value. Cannot be [null](#).

### **field** Field

The field that specifies which Autodesk.Revit.DB.XYZ value to retrieve. Cannot be [null](#).

## Returns

### XYZ

The Autodesk.Revit.DB.XYZ value associated with the specified field in the entity.

## Remarks

The method adapts to different versions of the API to ensure compatibility.

### Error messages:

- Entity or field cannot be null.
- Failed to retrieve the XYZ value from the entity.

## SetElementId(Entity, Field, ElementId)

Sets the specified ElementId value for the given Autodesk.Revit.DB.ExtensibleStorage.Entity and Autodesk.Revit.DB.ExtensibleStorage.Field.

```
public static void SetElementId(Entity entity, Field field, ElementId value)
```

## Parameters

**entity** Entity

The Autodesk.Revit.DB.ExtensibleStorage.Entity to which the value will be assigned. Cannot be [null](#).

**field** Field

The Autodesk.Revit.DB.ExtensibleStorage.Field that identifies the property to set. Cannot be [null](#).

**value** ElementId

The ElementId value to assign to the specified field. Cannot be [null](#).

## Remarks

The method ensures compatibility with different versions of the API by using the appropriate unit type when setting the value.

### Error messages:

- Entity, field, or value cannot be null.
- Failed to set the ElementId value for the specified field.

## SetXYZ(Entity, Field, XYZ)

Sets the specified field of the given entity to the provided XYZ value, using the appropriate unit system.

```
public static void SetXYZ(Entity entity, Field field, XYZ value)
```

## Parameters

**entity** Entity

The entity whose field is to be updated. Cannot be null.

**field** Field

The field of the entity to set. Cannot be null.

**value** XYZ

The XYZ value to assign to the specified field. Cannot be null.

## Remarks

The method determines the appropriate unit system based on the runtime environment and applies it when setting the field.

### Error messages:

- Entity, field, or value cannot be null.
- Failed to set the XYZ value for the specified field.

# Class GeometryUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Utility methods for geometry operations

```
public static class GeometryUtils
```

## Inheritance

[object](#) ← GeometryUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### CheckLineIntersectsCurves(List<Line>, Line)

```
public static bool CheckLineIntersectsCurves(List<Line> curves, Line testLine)
```

#### Parameters

curves [List](#)<Line>

testLine Line

#### Returns

[bool](#)

### CreateOffsetLine(Line, XYZ, double)

```
public static Line CreateOffsetLine(Line baseLine, XYZ direction, double offset)
```

## Parameters

**baseLine** Line

**direction** XYZ

**offset** double ↗

## Returns

Line

# Class MbCeilingUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with Revit ceiling elements.

```
public static class MbCeilingUtils
```

## Inheritance

[object](#) ← MbCeilingUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### IsPointOnCeiling(Ceiling, XYZ)

Determines whether a specified point lies on the bottom face of a given ceiling element.

```
public static MbResult<bool> IsPointOnCeiling(Ceiling ceiling, XYZ point)
```

#### Parameters

**ceiling** Ceiling

The Ceiling element to check against.

**point** XYZ

The Autodesk.Revit.DB.XYZ point to test for intersection with the ceiling's bottom face.

#### Returns

[MbResult<bool](#)

An MbResult<bool> indicating whether the point is on the ceiling's bottom face. Returns a successful result with `true` if the point projects onto the bottom face; otherwise, `false`. Returns a failed result if the ceiling or point is null, or if the bottom face cannot be determined.

# Class MbConnectorUtilities

Namespace: [MBF.Revit.Data\\_Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for managing and interacting with MEP connectors in a Revit document.

```
public static class MbConnectorUtilities
```

## Inheritance

[object](#) ← MbConnectorUtilities

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class focuses on general MEP connector operations, fitting analysis, and connector traversal. For pipe-specific operations, use [MbPipesUtils](#).

## Methods

### BypassNonConnectorFittings(FamilyInstance, List<ElementId>)

Traverses a chain of connected fittings starting from the specified fitting, bypassing any fittings that are of the type [ModelingHelperNonConnector](#).

```
public static MbResult<FamilyInstance> BypassNonConnectorFittings(FamilyInstance  
startingFitting, List<ElementId> excludedElementIds)
```

#### Parameters

**startingFitting** FamilyInstance

The initial fitting from which to start the traversal. Cannot be null.

**excludedElementIds** [List](#)<ElementId>

A list of element IDs to exclude from the traversal. Cannot be null.

## Returns

[MbResult](#)<FamilyInstance>

A [MbResult<T>](#) containing the first fitting that is not of type [ModelingHelperNonConnector](#), or an error message if the traversal fails.

## Remarks

The method will return a failure result if the `startingFitting` is null, if the `excludedElementIds` list is null, or if no suitable fitting is found within the maximum number of iterations. The traversal will also fail if any fitting in the chain cannot be processed due to missing family information or connectivity issues.

## GetConnectedElementsFromFitting(FamilyInstance)

Retrieves a list of elements that are directly or indirectly connected to the specified fitting.

```
public static MbResult<List<Element>> GetConnectedElementsFromFitting(FamilyInstance  
fitting)
```

## Parameters

**fitting** FamilyInstance

The FamilyInstance representing the fitting from which to find connected elements. Must not be null and must have a valid MEP model.

## Returns

[MbResult](#)<[List](#)<Element>>

An [MbResult<T>](#) containing a list of Element objects that are connected to the fitting. Returns a failure result if the fitting or its MEP model is null or invalid, or if an error occurs while retrieving connectors.

## Remarks

The method first attempts to find elements directly connected to the fitting's connectors. If a connector is not directly connected to an element, the method searches for the nearest unconnectTOed connector in the document.

## GetConnectorsFromFamilyInstance(FamilyInstance)

Retrieves connectors from the specified family instance.

```
public static IEnumerable<Connector> GetConnectorsFromFamilyInstance(FamilyInstance  
familyInstance)
```

Parameters

**familyInstance** FamilyInstance

The family instance from which to extract connectors.

Returns

[IEnumerable](#) <Connector>

An enumerable collection of connectors.

## GetConnectorsFromFlexPipe(FlexPipe)

Retrieves connectors from the specified flex pipe.

```
public static IEnumerable<Connector> GetConnectorsFromFlexPipe(FlexPipe flexPipe)
```

Parameters

**flexPipe** FlexPipe

The flex pipe from which to extract connectors.

Returns

[IEnumerable](#) <Connector>

An enumerable collection of connectors.

## GetDirectlyConnectedElement(Connector)

Retrieves the first element directly connected to the specified connector, excluding piping systems.

```
public static Element GetDirectlyConnectedElement(Connector connector)
```

## Parameters

**connector** Connector

The connector for which to find a directly connected element.

## Returns

Element

The first directly connected Element that is not a Autodesk.Revit.DB.Plumbing.PipingSystem, or [null](#) if no such element is found.

## GetRoundConnectorsFromFitting(FamilyInstance)

Retrieves a list of round connectors from the specified fitting.

```
public static MbResult<List<Connector>> GetRoundConnectorsFromFitting(FamilyInstance  
fitting)
```

## Parameters

**fitting** FamilyInstance

The FamilyInstance representing the fitting from which to extract round connectors. Must not be null and must have valid connectors.

## Returns

[MbResult<List><Connector>>](#)

An [MbResult<T>](#) containing a list of Connector objects that are round and not part of a piping system.

Returns a failure result if the fitting's connectors are null or invalid, or if no round connectors are found.

## IsMEPCategory(FamilyInstance)

Determines whether the specified family instance belongs to an MEP category.

```
public static bool IsMEPCategory(FamilyInstance familyInstance)
```

## Parameters

**familyInstance** FamilyInstance

The family instance to check.

## Returns

[bool](#)

True if the family instance belongs to an MEP category; otherwise, false.

# Class MbCurveUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Utility methods for working with curves in Revit.

```
public static class MbCurveUtils
```

## Inheritance

[object](#) ← MbCurveUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### EnforceChainForCurves(List<Curve>)

Ensures curves are chained together by connecting their endpoints.

```
public static MbResult<List<Curve>> EnforceChainForCurves(List<Curve> curves)
```

#### Parameters

curves [List](#)<Curve>

List of curves to chain.

#### Returns

[MbResult](#)<[List](#)<Curve>>

Success with a connected chain of curves, or Failure with an error message.

# Class MbFaceUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Retrieves the fill pattern associated with the surface material of the specified face.

```
public static class MbFaceUtils
```

## Inheritance

[object](#) ← MbFaceUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This method checks the material associated with the face and retrieves its surface foreground pattern. If the face does not have a valid material, or if the material does not have a surface foreground pattern, the method returns a failure result.

## Methods

### GetFillPattern(Face, Document)

Retrieves the surface foreground fill pattern associated with the material of the specified face.

```
public static MbResult<FillPatternElement> GetFillPattern(this Face face, Document document)
```

#### Parameters

**face** Face

The face for which to retrieve the fill pattern. Cannot be [null](#).

**document** Document

The document containing the face and its associated elements. Cannot be [null](#).

## Returns

[MbResult<FillPatternElement>](#)

A [MbResult<T>](#) containing the fill pattern element if successful, or a failure result with an error message if the operation cannot be completed.

## Remarks

This method checks the material associated with the specified face and retrieves its surface foreground fill pattern. If the face does not have a valid material, or if the material does not have a surface foreground pattern, the method returns a failure result.

# Class MbFamilyInstanceUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with family instances in a Revit model.

```
public static class MbFamilyInstanceUtils
```

## Inheritance

[object](#) ← MbFamilyInstanceUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

The [MbFamilyInstanceUtils](#) class offers methods to retrieve family names and types from Revit elements, ensuring that the elements are valid and belong to a proper document context.

## Methods

### GetElementFamilyName(ElementType)

Retrieves the family name of the specified element type.

```
public static MbResult<string> GetElementFamilyName(this ElementType elementType)
```

#### Parameters

**elementType** ElementType

The element type from which to obtain the family name. Must not be null and must be a valid object.

#### Returns

[MbResult<string>](#)

An [MbResult<T>](#) containing the family name of the element type if successful; otherwise, a failure result with an error message.

## GetElementType(Element)

Retrieves the ElementType of the specified `element`.

```
public static MbResult<ElementType> GetElementType(this Element element)
```

### Parameters

`element` ElementType

The element for which to retrieve the type. Must not be null and must belong to a valid document.

### Returns

[MbResult<ElementType>](#)

An [MbResult<T>](#) containing the ElementType if successful; otherwise, a failure result with an error message.

### Remarks

This method returns a failure result if the element is null, belongs to a family document, has no valid type, or if the retrieved type is not valid.

## GetElementTypeName(ElementType)

Retrieves the name of the specified ElementType.

```
public static MbResult<string> GetElementTypeName(this ElementType elementType)
```

### Parameters

`elementType` ElementType

The ElementType whose name is to be retrieved. Must not be null and must be valid.

### Returns

## [MbResult<string>](#)

An [MbResult<T>](#) containing the name of the `elementType` if successful; otherwise, an [MbResult<T>](#) indicating failure with an appropriate error message.

## GetFamilyNameAndType(Element)

Retrieves the family name and type of the specified element.

```
public static MbResult<string> GetFamilyNameAndType(this Element element)
```

### Parameters

`elementType` Element

The element from which to retrieve the family name and type. Cannot be null and must be a valid object.

### Returns

[MbResult<string>](#)

A [MbResult<T>](#) containing a string formatted as "FamilyName-TypeName" if successful; otherwise, an error message indicating the failure reason.

### Remarks

This method checks the validity of the provided element and retrieves its type and family names. If the element is invalid or any retrieval operation fails, the method returns a failure result with an appropriate error message.

# Class MbFlexPipesUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Filters the provided collection of Revit elements and retrieves all flex pipes.

```
public static class MbFlexPipesUtils
```

## Inheritance

[object](#) ← MbFlexPipesUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This method ensures that only elements categorized as flex pipes ( `BuiltInCategory.OST_FlexPipeCurves` ) are returned. It uses the category information of each element to perform the filtering.

## Methods

### GetFlexPipeInstances(IEnumerable<Element>)

Filters the provided collection of Revit elements and retrieves all flex pipes.

```
public static IEnumerable<FlexPipe> GetFlexPipeInstances(IEnumerable<Element>  
selectedElements)
```

#### Parameters

**selectedElements** [IEnumerable](#)<Element>

A collection of Revit elements to filter.

#### Returns

[IEnumerable](#)<FlexPipe>

An [IEnumerable<T>](#) of Autodesk.Revit.DB.Plumbing.FlexPipe objects.

## Remarks

This method ensures that only elements categorized as flex pipes (BuiltInCategory.OST\_FlexPipeCurves) are returned.

Possible error messages:

- **ArgumentNullException:** Thrown if `selectedElements` is null.
- **InvalidOperationException:** Thrown if an element cannot be cast to Autodesk.Revit.DB.Plumbing.FlexPipe.

# Class MbLinkedElementUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with linked elements in Revit documents, including coordinate transformations between parent and linked files.

```
public static class MbLinkedElementUtils
```

## Inheritance

[object](#) ← MbLinkedElementUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### GetPointInLinkFileFromParentFile(XYZ, RevitLinkInstance, Document, bool)

Converts a point from the parent Revit document's coordinate system to the linked file's coordinate system.

```
public static MbResult<XYZ> GetPointInLinkFileFromParentFile(XYZ pointInParent,  
RevitLinkInstance linkInstance, Document parentDocument, bool sharedPositioningEnabled  
= false)
```

#### Parameters

**pointInParent** XYZ

The Autodesk.Revit.DB.XYZ point in the parent document's coordinate system.

**linkInstance** RevitLinkInstance

The Autodesk.Revit.DB.RevitLinkInstance representing the linked Revit file.

**parentDocument** Document

The parent Autodesk.Revit.DB.Document containing the original point.

**sharedPositioningEnabled** [bool](#)

If **true**, uses shared positioning based on survey points; otherwise, uses the link's transform.

Returns

[MbResult<XYZ>](#)

An [MbResult<T>](#) containing the transformed point in the linked file's coordinate system, or an error message if the transformation fails.

Remarks

This method first attempts to use shared positioning if enabled, shifting the point by the difference between the parent and link survey points. If shared positioning is not enabled, it applies the inverse of the link's transform to the point after transforming it by the parent document's transform.

## GetPointInParentFileFromLink(XYZ, RevitLinkInstance, Document, bool)

Converts a point from the linked Revit file's coordinate system to the parent document's coordinate system.

```
public static MbResult<XYZ> GetPointInParentFileFromLink(XYZ pointInLink, RevitLinkInstance  
revitLinkInstance, Document parentDoc, bool sharedPositioningEnabled = false)
```

Parameters

**pointInLink** XYZ

The Autodesk.Revit.DB.XYZ point in the linked file's coordinate system.

**revitLinkInstance** RevitLinkInstance

The Autodesk.Revit.DB.RevitLinkInstance representing the linked Revit file.

**parentDoc** Document

The parent Autodesk.Revit.DB.Document to which the point will be transformed.

## `sharedPositioningEnabled` [bool](#)

If `true`, uses shared positioning based on survey points; otherwise, uses the link's transform.

## Returns

### [MbResult<XYZ>](#)

An [MbResult<T>](#) containing the transformed point in the parent document's coordinate system, or an error message if the transformation fails.

## Remarks

This method first attempts to use shared positioning if enabled, shifting the point by the difference between the parent and link survey points. If shared positioning is not enabled, it applies the inverse of the parent document's transform to the point after transforming it by the link's transform.

# Class MbParameterUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with Revit parameters.

```
public static class MbParameterUtils
```

## Inheritance

[object](#) ← MbParameterUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### GetParameterValue(StorageType, Parameter)

Retrieves the string representation of a parameter's value based on its Autodesk.Revit.DB.StorageType.

```
public static string GetParameterValue(StorageType storageType, Parameter instanceParameter)
```

#### Parameters

**storageType** StorageType

The storage type of the parameter (e.g. Double, Integer, String, ElementId).

**instanceParameter** Parameter

The parameter from which to extract the value.

#### Returns

[string](#)

A string representation of the parameter's value.

Possible error messages:

- "**N/A**" - Returned if the parameter value is a string and is **null**.
- "**Unsupported Type**" - Returned if the storage type is not supported.

# Class MbPipeAccessoriesUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with pipe accessory elements in Revit.

```
public static class MbPipeAccessoriesUtils
```

## Inheritance

[object](#) ← MbPipeAccessoriesUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class contains methods to filter and retrieve Revit elements categorized as pipe accessories. It is designed to work with collections of Revit elements and supports conditional compilation for compatibility with different Revit versions.

## Methods

### GetPipeAccessoriesInstances(IEnumerable<Element>)

Retrieves all FamilyInstance elements categorized as pipe accessories from the provided collection of Revit elements.

```
public static IEnumerable<FamilyInstance> GetPipeAccessoriesInstances(IEnumerable<Element> selectedElements)
```

#### Parameters

**selectedElements** [IEnumerable](#)<Element>

A collection of Element objects to search within.

#### Returns

## IEnumerable <FamilyInstance>

An [IEnumerable<T>](#) containing all elements from `selectedElements` that are instances of families and belong to the `BuiltInCategory.OST_PipeAccessory` category.

## Remarks

This method filters the input collection to include only elements whose category is "Pipe Accessories", then casts those elements to `FamilyInstance` for further use. The filtering logic uses conditional compilation to support different Revit versions.

Possible error messages:

- **ArgumentNullException:** Thrown if `selectedElements` is null.
- **InvalidCastException:** Thrown if an element cannot be cast to `FamilyInstance` (should not occur due to `OfType` usage).

# Class MbPipeFittingsUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with pipe fittings in Revit.

```
public static class MbPipeFittingsUtils
```

## Inheritance

[object](#) ← MbPipeFittingsUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### FittingSizeIsEqual(FamilyInstance)

Checks if all connectors of a given fitting have equal sizes (radius).

```
public static MbResult<bool> FittingSizeIsEqual(FamilyInstance instance)
```

#### Parameters

**instance** FamilyInstance

The FamilyInstance of the fitting to check.

#### Returns

[MbResult<bool>](#)

Returns an [MbResult<T>](#) where T is [bool](#):

- If [IsSuccess](#) is [true](#), the operation completed successfully.
- If [Value](#) is [true](#), all connectors have equal sizes or there are fewer than two connectors.
- If [Value](#) is [false](#), connectors have different sizes.
- If [IsSuccess](#) is [false](#), an error occurred, and [ErrorMessage](#) provides details:

- "Instance is null."

## GetAngleBetweenLargestRadiusConnectors(FamilyInstance)

Calculates the angle between the two connectors with the largest radius in the given FamilyInstance.

```
public static MbResult<double> GetAngleBetweenLargestRadiusConnectors(FamilyInstance instance)
```

### Parameters

**instance** FamilyInstance

The FamilyInstance containing connectors to analyze.

### Returns

[MbResult<double>](#)

Returns an [MbResult<T>](#) where T is **double**:

- If **IsSuccess** is **true**, the operation completed successfully.
- If **Value** is a valid angle, it represents the computed angle between the two connectors with the largest radius.
- If **IsSuccess** is **false**, an error occurred, and **ErrorMessage** provides details:
  - "FamilyInstance is null."

## GetAngleBetweenLargestRadiusConnectors(List<Connector>)

Calculates the angle between the two connectors with the largest radius in the given list of connectors.

```
public static MbResult<double> GetAngleBetweenLargestRadiusConnectors(List<Connector> connectors)
```

### Parameters

**connectors** [List<Connector>](#)

A list of connectors to analyze.

### Returns

## [MbResult<double>](#)

Returns an [MbResult<T>](#) where T is `double`:

- If `.IsSuccess` is `true`, the operation completed successfully.
- If `Value` is a valid angle, it represents the computed angle between the two connectors with the largest radius.
- If `.IsSuccess` is `false`, an error occurred, and `ErrorMessage` provides details:
  - "At least two connectors are required."

## GetElbowsFamilyInstances(IEnumerable<Element>)

Filters the provided collection of Revit elements and retrieves all pipe fitting family instances that are categorized as "Elbow" fittings.

```
public static IEnumerable<FamilyInstance> GetElbowsFamilyInstances(IEnumerable<Element> selectedElements)
```

### Parameters

`selectedElements` [IEnumerable<Element>](#)

A collection of Revit elements to filter.

### Returns

[IEnumerable<FamilyInstance>](#)

An [IEnumerable<T>](#) of `FamilyInstance` objects representing elbow fittings.

### Remarks

This method ensures that only elements categorized as pipe fittings (`BuiltInCategory.OST_PipeFitting`) and specifically of type `PartType.Elbow` are returned.

## GetFarthestConnectorSize(FamilyInstance, Pipe)

Returns the size (diameter) of the valid round connector that is farthest from the given pipe.

```
public static double? GetFarthestConnectorSize(FamilyInstance fitting, Pipe pipe)
```

## Parameters

**fitting** FamilyInstance

The outlet fitting instance.

**pipe** Pipe

The reference pipe to compare distance against.

## Returns

[double](#)?

The diameter (in internal units) of the farthest valid round connector; or null if none found.

## GetInlineAndBullNoseConnectors(List<Connector>)

Identifies and returns a tuple containing two connectors that are opposite in direction and a third connector.

```
public static (Connector, Connector, Connector)
GetInlineAndBullNoseConnectors(List<Connector> connectors)
```

## Parameters

**connectors** [List](#)<Connector>

A list of connectors to be evaluated. The list must contain at least three connectors.

## Returns

(Connector, Connector, Connector)

A tuple containing two connectors that are opposite in direction and a third connector, or [null](#) if no such pair is found.

## Remarks

The method assumes that the list contains at least three connectors and that there is exactly one pair of connectors with opposite directions. The third connector in the tuple is considered the "bull nose" connector.

## GetOrderedConnectorSizesInInch(Element, bool)

Retrieves a sorted list of connector sizes (diameters) in inches for a given Element that can be either an Autodesk.Revit.DB.MEPCurve or FamilyInstance.

```
public static MbResult<List<double>> GetOrderedConnectorSizesInInch(Element element, bool  
isAscending = true)
```

### Parameters

**element** Element

The Element representing an Autodesk.Revit.DB.MEPCurve or FamilyInstance.

**isAscending** [bool](#)

Specifies whether to sort the sizes in ascending (true) or descending (false) order.

### Returns

[MbResult<List<double>>](#)

An [MbResult<T>](#) where T is [List<T>](#):

- If **IsSuccess** is **true**, the operation completed successfully.
- The list contains the diameters of all connectors in inches, sorted in the specified order.
- If **IsSuccess** is **false**, an error occurred, and **ErrorMessage** provides details:
  - "Element is null."
  - "No connectors found."
  - "Connector list must contain exactly 3 items for a tee fitting."

## GetOrderedConnectorSizesInInch(FamilyInstance, bool)

Retrieves a sorted list of connector sizes (diameters) for a given fitting.

```
public static MbResult<List<double>> GetOrderedConnectorSizesInInch(FamilyInstance instance,  
bool isAscending = true)
```

### Parameters

**instance** FamilyInstance

The Autodesk.Revit.DB.FamilyInstance representing the fitting.

### `isAscending` `bool`

Specifies whether to sort the sizes in ascending (true) or descending (false) order.

### Returns

### `MbResult<List<double>>`

An `MbResult<T>` where `T` is `List<T>`:

- If `.IsSuccess` is `true`, the operation completed successfully.
- The list contains the diameters of all connectors in the specified order.
- If `.IsSuccess` is `false`, an error occurred and `ErrorMessage` provides details:
  - "Instance is null."
  - "No connectors found."

## GetOtherFittingWhichIsNotElbowsOrTeesOrOutletsFamilyInstances(IEnumerable<Element>)

Filters the provided collection of Revit elements and retrieves all pipe fitting family instances that are not categorized as "Tee", "Elbow", or "Outlets" fittings.

```
public static IEnumerable<FamilyInstance>
GetOtherFittingWhichIsNotElbowsOrTeesOrOutletsFamilyInstances(IEnumerable<Element>
selectedElements)
```

### Parameters

#### `selectedElements` `IEnumerable<Element>`

A collection of Revit elements to filter.

### Returns

#### `IEnumerable<FamilyInstance>`

An `IEnumerable<T>` of `FamilyInstance` objects representing fittings that are not tee, elbow, or outlet fittings.

### Remarks

This method ensures that only elements categorized as pipe fittings (`BuiltInCategory.OST_PipeFitting`) and excluding types `PartType.Tee`, `PartType.Elbow`, or `PartType.SpudAdjustable` are returned.

## GetOutletDirectionCode(Pipe, FamilyInstance)

Determines the outlet orientation code (e.g., 1 = Top, 2 = Right, etc.) relative to the pipe axis, based on job file standards.

```
public static double GetOutletDirectionCode(Pipe pipe, FamilyInstance outlet)
```

### Parameters

**pipe** Pipe

The host pipe.

**outlet** FamilyInstance

The outlet FamilyInstance.

### Returns

[double](#)

Directional code (1.0 = Top, 1.5 = Top-Right 45°, etc.), or -1 if undetermined.

- -1: No matching direction found within tolerance or outlet connector not found.

## GetOutletsFamilyInstances(IEnumerable<Element>)

Filters the provided collection of Revit elements and retrieves all pipe fitting family instances that are categorized as "Outlets" (Spud Adjustable) fittings.

```
public static IEnumerable<FamilyInstance> GetOutletsFamilyInstances(IEnumerable<Element> selectedElements)
```

### Parameters

**selectedElements** [IEnumerable](#)<Element>

A collection of Revit elements to filter.

### Returns

[IEnumerable](#)<FamilyInstance>

An [IEnumerable<T>](#) of FamilyInstance objects representing outlet fittings.

## Remarks

This method ensures that only elements categorized as pipe fittings (BuiltInCategory.OST\_PipeFitting) and specifically of type PartType.SpudAdjustable (representing outlets) are returned.

## GetPerpendicularClockwise(XYZ)

Returns a vector perpendicular to the input vector in the clockwise direction, on the XY plane.

```
public static XYZ GetPerpendicularClockwise(this XYZ vector)
```

### Parameters

**vector** XYZ

Input vector.

### Returns

XYZ

Clockwise perpendicular vector on XY plane.

## GetPerpendicularCounterClockwise(XYZ)

Returns a vector perpendicular to the input vector in the counter-clockwise direction, on the XY plane.

```
public static XYZ GetPerpendicularCounterClockwise(this XYZ vector)
```

### Parameters

**vector** XYZ

Input vector.

### Returns

XYZ

Counter-clockwise perpendicular vector on XY plane.

## GetPerpendicularPlaneAxes(XYZ)

Returns a tuple of two orthogonal vectors that define a plane perpendicular to the input vector: - First: a stable perpendicular vector (clockwise or counter-clockwise). - Second: the cross-product (normal) to establish a consistent plane direction.

```
public static (XYZ Perpendicular, XYZ Normal) GetPerpendicularPlaneAxes(this  
XYZ pipeDirection)
```

### Parameters

**pipeDirection** XYZ

The primary pipe direction vector.

### Returns

(XYZ [Perpendicular](#), XYZ [Normal](#))

A tuple (perpendicular, normal) defining the orientation plane.

## GetTeeFamilyInstances(IEnumerable<Element>)

Filters the provided collection of Revit elements and retrieves all pipe fitting family instances that are categorized as "Tee" fittings.

```
public static IEnumerable<FamilyInstance> GetTeeFamilyInstances(IEnumerable<Element>  
selectedElements)
```

### Parameters

**selectedElements** [IEnumerable](#)<Element>

A collection of Revit elements to filter.

### Returns

[IEnumerable](#)<FamilyInstance>

An [IEnumerable<T>](#) of FamilyInstance objects representing tee fittings.

## Remarks

This method ensures that only elements categorized as pipe fittings (BuiltInCategory.OST\_PipeFitting) and specifically of type PartType.Tee are returned.

## GetValidOutletConnectors(FamilyInstance)

Returns only valid round connectors from the outlet fitting.

```
public static List<Connector> GetValidOutletConnectors(FamilyInstance fitting)
```

## Parameters

**fitting** FamilyInstance

The outlet fitting instance.

## Returns

[List](#)<Connector>

A list of connectors that are round and not owned by a Autodesk.Revit.DB.Plumbing.PipingSystem.

# Class MbPipesUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with pipe elements in Revit.

```
public static class MbPipesUtils
```

## Inheritance

[object](#) ← MbPipesUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### ArePointsClose(FamilyInstance, FamilyInstance, double)

Determines whether the locations of two FamilyInstance objects are within a specified distance threshold.

```
public static bool ArePointsClose(FamilyInstance a, FamilyInstance b, double threshold)
```

#### Parameters

**a** FamilyInstance

The first FamilyInstance. Must not be [null](#).

**b** FamilyInstance

The second FamilyInstance. Must not be [null](#).

**threshold** [double](#)

The maximum distance, in the same units as the points' coordinates, within which the two instances are considered close.

Returns

[bool](#)

[true](#) if the distance between the locations of [a](#) and [b](#) is less than [threshold](#); otherwise, [false](#).

Remarks

If either [a](#) or [b](#) is [null](#), or if their locations cannot be determined, the method returns [false](#).

## ArePointsSame(XYZ, XYZ)

Determines whether two points in 3D space are considered the same based on a small tolerance.

```
public static bool ArePointsSame(XYZ point1, XYZ point2)
```

Parameters

[point1](#) XYZ

The first point to compare.

[point2](#) XYZ

The second point to compare.

Returns

[bool](#)

[true](#) if the distance between [point1](#) and [point2](#) is less than 0.01; otherwise, [false](#).

Remarks

This method uses a fixed tolerance of 0.01 to account for minor differences in floating-point calculations.

## FamilyIsOutsideLine(FamilyInstance, Line)

Determines whether the specified family instance is outside the given line.

```
public static bool FamilyIsOutsideLine(FamilyInstance familyInstance, Line line)
```

## Parameters

### **familyInstance** FamilyInstance

The family instance to evaluate. Must have a valid Autodesk.Revit.DB.LocationPoint.

### **line** Line

The line against which the family instance's position is evaluated.

## Returns

### bool

true if the family instance is projected onto one of the endpoints of the line; otherwise, false.

## Remarks

This method checks if the projection of the family instance's location point onto the specified line coincides with either of the line's endpoints. If the family instance does not have a valid Autodesk.Revit.DB.LocationPoint , the method returns false.

## FindNearbyUnconnectedConnector(XYZ, Document, List<ElementId>, HashSet<int>, string)

Finds the nearest unconnected connector to a specified point within a given document.

```
public static Connector FindNearbyUnconnectedConnector(XYZ targetPoint, Document doc,
List<ElementId> excludedElementIds, HashSet<int> discoveredElementIds = null, string caller
= null)
```

## Parameters

### **targetPoint** XYZ

The target point in 3D space to search for nearby connectors.

### **doc** Document

The Revit document containing the elements to search.

### **excludedElementIds** List<ElementId>

A list of element IDs to exclude from the search.

#### **discoveredElementIds** [HashSet<int>](#)

An optional set of element IDs that have already been processed. These elements will be excluded from the search.

#### **caller** [string](#)

The name of the calling method, automatically provided by the compiler. Used to adjust behavior for specific callers.

Returns

Connector

The nearest unconnected Connector to the [targetPoint](#), or [null](#) if no suitable connector is found.

Remarks

This method searches for connectors within a bounding box centered on the [targetPoint](#). It filters out connectors belonging to elements that are excluded or already processed, as well as connectors whose owners are subcomponents of other elements. The method prioritizes connectors based on their proximity to the [targetPoint](#) and returns the closest one within its connection radius.

## GetConnectedFittingsBothSidesOfPipe(Document, Pipe)

Retrieves the connectors of fittings connected to both ends of a specified pipe.

```
public static List<Connector> GetConnectedFittingsBothSidesOfPipe(Document doc, Pipe pipe)
```

Parameters

#### **doc** Document

The Revit document containing the pipe and its connected elements.

#### **pipe** Pipe

The pipe for which to find connected fittings on both ends.

Returns

## [List](#) <Connector>

A list of Connector objects representing the connectors of fittings connected to both ends of the specified pipe. The list will be empty if no fittings are connected.

## Remarks

This method identifies fittings connected to the pipe's end connectors and includes only those fittings that are spuds, tees, or elbows. If no fittings are found, the method attempts to locate connectors based on the pipe's nearest endpoints.

## GetConnectedFittingsOnCurve(Pipe)

Retrieves a list of fittings connected to the specified pipe along its curve connectors.

```
public static List<FamilyInstance> GetConnectedFittingsOnCurve(this Pipe pipe)
```

## Parameters

**pipe** Pipe

The pipe for which to find connected fittings. Cannot be [null](#).

## Returns

## [List](#) <FamilyInstance>

A list of FamilyInstance objects representing the fittings connected to the pipe. The list will be empty if no fittings are connected.

## Remarks

This method examines the curve-type connectors of the specified pipe and identifies any connected fittings. Only fittings directly connected to the pipe's curve connectors are included in the result.

## GetConnectorsForElement(Element)

Retrieves a list of valid connectors associated with the specified element.

```
public static List<Connector> GetConnectorsForElement(Element element)
```

## Parameters

### **element** Element

The element for which to retrieve the connectors. Must not be [null](#).

## Returns

### [List](#)<Connector>

A list of Connector objects that are valid and associated with the specified element. If no valid connectors are found, an empty list is returned.

## GetEndConnectors(Pipe)

Returns the end connectors of a pipe, excluding those owned by a Autodesk.Revit.DB.Plumbing.Piping System.

```
public static List<Connector> GetEndConnectors(this Pipe pipe)
```

## Parameters

### **pipe** Pipe

The pipe to evaluate.

## Returns

### [List](#)<Connector>

A list of end Connector objects not owned by a piping system.

## GetEndPoint(Pipe, PipePointType)

Retrieves the specified endpoint of the pipe.

```
public static XYZ GetEndPoint(this Pipe pipe, PipePointType pipePointType)
```

## Parameters

### **pipe** Pipe

The pipe from which to retrieve the endpoint.

#### **pipePointType** [PipePointType](#)

Specifies whether to retrieve the start or end point of the pipe.

Returns

XYZ

The endpoint of the pipe as specified by **pipePointType**.

## GetFarSideConnectedToFamilyInstance(Document, Pipe, XYZ)

Gets the connector on the far end of the pipe (opposite to the reference point) that is connected to a FamilyInstance.

```
public static Connector GetFarSideConnectedToFamilyInstance(Document doc, Pipe pipe,  
XYZ referencePoint)
```

Parameters

#### **doc** Document

The Revit document.

#### **pipe** Pipe

The pipe to evaluate.

#### **referencePoint** XYZ

A point used to identify the far end of the pipe.

Returns

Connector

The connected Connector on the far side, or null if not found.

## GetFittingAtFarthestConnector(Pipe, XYZ)

Finds the fitting connected to the farthest connector of a pipe from the given start point.

```
public static MbResult<Element> GetFittingAtFarthestConnector(Pipe pipe, XYZ startPoint)
```

## Parameters

**pipe** Pipe

The pipe element to inspect.

**startPoint** XYZ

The reference point to compare connector distances to.

## Returns

[MbResult<Element>](#)

A containing the connected fitting element, or a failure message if not found.

## GetFittingProjectedPoint(FamilyInstance, Line)

Projects the location point of a fitting onto a specified pipeline.

```
public static XYZ GetFittingProjectedPoint(FamilyInstance fitting, Line pipeline)
```

## Parameters

**fitting** FamilyInstance

The fitting whose location point is to be projected. Must be a FamilyInstance with a Autodesk.Revit.DB.LocationPoint.

**pipeline** Line

The line representing the pipeline onto which the fitting's point will be projected.

## Returns

XYZ

The projected point on the pipeline if the fitting has a location point; otherwise, [null](#).

## GetFittingStation(FamilyInstance, Line, XYZ)

Calculates the distance from the start point of a pipeline to the projection of a fitting's location onto the pipeline.

```
public static double GetFittingStation(FamilyInstance fitting, Line pipeline,  
XYZ pipeStartPoint)
```

### Parameters

**fitting** FamilyInstance

The fitting whose location is to be projected onto the pipeline. Must have a valid location point.

**pipeline** Line

The line representing the pipeline onto which the fitting's location will be projected.

**pipeStartPoint** XYZ

The starting point of the pipeline from which the distance is measured.

### Returns

[double](#)

The distance from the **pipeStartPoint** to the projected point of the fitting's location on the pipeline.

Returns 0.0 if the fitting does not have a valid location point.

## GetFittingsOnPipe(Document, Pipe, XYZ)

Retrieves a list of fitting element IDs that are associated with the specified pipe.

```
public static List<ElementId> GetFittingsOnPipe(Document doc, Pipe pipe, XYZ sortingPoint  
= null)
```

### Parameters

**doc** Document

The Revit document containing the pipe and fittings.

**pipe** Pipe

The pipe for which to find associated fittings. Must be a valid pipe element.

#### **sortingPoint** XYZ

An optional point used to sort the fittings by proximity. If [null](#), the fittings are not sorted.

#### Returns

#### [List](#)<ElementId>

A list of ElementId objects representing the fittings associated with the pipe. The list may be empty if no fittings are found.

#### Remarks

This method identifies fittings associated with the pipe based on several criteria, including:

- Fittings that intersect the pipe's solid geometry.
- Fittings that are within a proximity threshold of the pipe's centerline.
- Fittings that are well-connected to the pipe's curve.

Duplicate fittings are removed, and fittings at the pipe's endpoints are excluded from the result. If a **sortingPoint** is provided, the fittings are sorted by their proximity to the point.

## GetFittingsOnPipeByStation(Document, Pipe, XYZ, double)

Retrieves a dictionary of fittings grouped by their station positions along a specified pipe.

```
public static Dictionary<double, List<FamilyInstance>> GetFittingsOnPipeByStation(Document doc, Pipe pipe, XYZ pipeStartPoint, double stationTolerance = 0.164)
```

#### Parameters

##### **doc** Document

The document containing the pipe and fittings.

##### **pipe** Pipe

The pipe for which fittings are to be retrieved and grouped.

##### **pipeStartPoint** XYZ

The starting point of the pipe used to calculate station positions.

## **stationTolerance** [double](#)

The tolerance for grouping fittings at the same station, in the same units as the pipe's length. Default is 0.164.

## Returns

[Dictionary](#)<[double](#), [List](#)<FamilyInstance>>

A dictionary where each key is a station position (distance from the pipe's start point) and the value is a list of fittings located at that station. The dictionary will be empty if no valid fittings are found.

## Remarks

This method identifies fittings that intersect with the pipe's solid geometry, are in close proximity to the pipe, or are well-connected along the pipe's curve. Fittings at the pipe's endpoints are excluded from the results. The method uses a specified tolerance to group fittings that are close to each other into the same station.

## GetLineOfPipe(Pipe)

Retrieves the geometric line representation of the specified pipe.

```
public static Line GetLineOfPipe(this Pipe pipe)
```

## Parameters

**pipe** Pipe

The pipe for which to retrieve the line representation. Cannot be [null](#).

## Returns

Line

A Autodesk.Revit.DB.Line object representing the pipe's geometry. If the pipe has a stored start point, the line is created using the start point and the nearest endpoint. Otherwise, the line is created using the pipe's start and end points.

## Remarks

This method determines the line representation of a pipe based on its stored start point, if available, or its defined start and end points. The returned line can be used for geometric calculations or visualization.

## GetPipeCurve(Pipe)

Retrieves the geometric curve that defines the center line of the specified pipe.

```
public static Curve GetPipeCurve(this Pipe pipe)
```

### Parameters

**pipe** Pipe

The pipe from which to retrieve the center line curve. Must not be `.`.

### Returns

Curve

The representing the center line of the pipe, or if the pipe does not have a valid location curve.

### Remarks

This method extracts the form the of the pipe. Ensure that the pipe has a valid location curve before calling this method.

## GetPipeInstances(IEnumerable<Element>)

Retrieves all pipe instances from the specified collection of Revit elements.

```
public static IEnumerable<Pipe> GetPipeInstances(IEnumerable<Element> pipeSelectedElements)
```

### Parameters

**pipeSelectedElements** [IEnumerable](#)<Element>

A collection of Revit elements to filter for pipe instances.

### Returns

[IEnumerable](#)<Pipe>

An enumerable collection of Autodesk.Revit.DB.Plumbing.Pipe objects representing the pipe instances found in the input collection. If no pipe instances are found, the returned collection will be empty.

## Remarks

This method filters the provided elements to include only those that belong to the "Pipe Curves" category and are of type Autodesk.Revit.DB.Plumbing.Pipe. The filtering logic adapts to the Revit version being used.

## GetValidSpudFittings(Document)

Retrieves a list of valid spud fittings from the specified document.

```
public static List<FamilyInstance> GetValidSpudFittings(this Document doc)
```

## Parameters

**doc** Document

The Revit document to search for spud fittings.

## Returns

[List](#)<FamilyInstance>

A list of FamilyInstance objects representing valid spud fittings in the document. The list will be empty if no valid spud fittings are found.

## Remarks

This method filters elements in the document to include only pipe fittings that are instances of Family Instance and meet specific criteria for validity. Elements that do not pass the validity check are excluded.

## IsPointOnPipeEndPoints(XYZ, Pipe)

Determines whether the specified point is located on one of the endpoints of the given pipe.

```
public static bool IsPointOnPipeEndPoints(XYZ point, Pipe pipe)
```

## Parameters

**point** XYZ

The point to check.

**pipe** Pipe

The pipe whose endpoints are to be evaluated.

Returns

[bool](#)

[true](#) if the specified point is on one of the pipe's endpoints; otherwise, [false](#) .

## IsSpudAtEndPointsOfPipe(Pipe, ElementId, Document)

Determines whether the specified element is located at either endpoint of the given pipe.

```
public static bool IsSpudAtEndPointsOfPipe(Pipe pipe, ElementId elementId, Document doc)
```

Parameters

**pipe** Pipe

The pipe to check for endpoint connections.

**elementId** ElementId

The ID of the element to evaluate.

**doc** Document

The document containing the pipe and element.

Returns

[bool](#)

[true](#) if the element is located at one of the endpoints of the pipe; otherwise, [false](#) .

## IsSpudFitting(FamilyInstance)

Determines whether the specified FamilyInstance represents a spud fitting.

```
public static bool IsSpudFitting(FamilyInstance instance)
```

## Parameters

**instance** FamilyInstance

The FamilyInstance to evaluate. Must not be [null](#).

## Returns

[bool](#)

[true](#) if the **instance** is a spud fitting; otherwise, [false](#).

## Remarks

A spud fitting is identified as a FamilyInstance with no FamilyInstance.SuperComponent and an Autodesk.Revit.DB.MEPModel of type Autodesk.Revit.DB.Mechanical.MechanicalFitting where the Autodesk.Revit.DB.Mechanical.MechanicalFitting.PartType is either PartType.SpudAdjustable or PartType.SpudPerpendicular .

## IsTeeOrElbowFitting(FamilyInstance)

Determines whether the specified FamilyInstance represents a tee or elbow fitting.

```
public static bool IsTeeOrElbowFitting(FamilyInstance instance)
```

## Parameters

**instance** FamilyInstance

The FamilyInstance to evaluate. This parameter can be [null](#).

## Returns

[bool](#)

[true](#) if the **instance** is a mechanical fitting of type tee or elbow; otherwise, [false](#).

## Remarks

This method checks if the `instance` has an Autodesk.Revit.DB.MEPModel of type Autodesk.Revit.DB.Mechanical.MechanicalFitting and evaluates its PartType to determine if it is a tee or elbow fitting.

## IsWellConnected(FamilyInstance)

Determines whether the specified FamilyInstance is well-connected.

```
public static bool IsWellConnected(FamilyInstance instance)
```

### Parameters

`instance` FamilyInstance

The FamilyInstance to evaluate. This parameter cannot be [null](#).

### Returns

[bool](#)

[true](#) if all connectors in the `instance`'s MEP model are connected; otherwise, [false](#). Returns [false](#) if the `instance` or its MEP model is [null](#).

## RemoveDuplicateFittings(ref List<FamilyInstance>)

Removes duplicate fittings from the provided list based on proximity and connection criteria.

```
public static void RemoveDuplicateFittings(ref List<FamilyInstance> fittings)
```

### Parameters

`fittings` [List](#)<FamilyInstance>

A reference to the list of FamilyInstance objects representing the fittings. The list will be modified to contain only unique fittings after the method completes.

### Remarks

Two fittings are considered duplicates if their positions are within a predefined threshold. When duplicates are found, the method retains the fitting with the most connected connectors, or the first fitting in the group if no fitting is better connected.

# SortFittingsByProximity(Document, List<ElementId>, XYZ)

Groups and sorts a list of fitting elements by their proximity to a specified reference point.

```
public static List<List<ElementId>> SortFittingsByProximity(Document doc, List<ElementId> fittingIds, XYZ referencePoint)
```

## Parameters

**doc** Document

The Revit document containing the fitting elements.

**fittingIds** [List](#)<ElementId>

A list of ElementId objects representing the fittings to be sorted.

**referencePoint** XYZ

The reference point used to calculate the proximity of each fitting.

## Returns

[List](#)<[List](#)<ElementId>>

A list of grouped lists of ElementId objects, where each inner list contains fittings that are within a specified proximity tolerance of each other. The groups are sorted by the distance of their first element to the reference point.

## Remarks

The method groups fittings that are within 5 centimeters (0.164 feet) of each other and sorts the groups based on the distance of the first fitting in each group to the reference point. This can be useful for organizing fittings spatially in a Revit model.

# Class MbProjectLocationUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with project location transformations in Revit.

```
public static class MbProjectLocationUtils
```

## Inheritance

[object](#) ← MbProjectLocationUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes methods to calculate transformations between the project base point and survey point, as well as retrieve project position data. It is intended for use in scenarios where precise location and orientation data are required, such as aligning models or extracting geospatial information.

## Methods

### GetProjectLocationTransform(Document)

Retrieves the transformation matrix and survey point coordinates that define the relationship between the project's internal origin and its shared coordinates system.

```
public static MbResult<(Transform Transform, XYZ SurveyPoint)>
GetProjectLocationTransform(Document document)
```

#### Parameters

**document** Document

The Revit document from which to retrieve the project location transform. Cannot be [null](#).

#### Returns

## [MbResult](#)<(Transform [Transform](#), XYZ [SurveyPoint](#))>

A result containing a tuple with the transformation matrix and the survey point coordinates. If the operation fails, the result will contain an error message describing the reason for failure.

## Remarks

The transformation matrix combines translation and rotation to align the internal origin with the shared coordinates system. The survey point coordinates represent the shared survey point in the project's coordinate system.

### Possible error messages returned to the user:

- "Document is null." - The provided `document` argument is [null](#). This indicates that the method was called without a valid Revit document instance.
- "Unable to retrieve project position." - The active project location or its position could not be obtained. This may occur if the project location is not set or is invalid in the current document.
- Any error message returned by [GetSharedSurveyPointCoordinates\(Document\)](#). These errors are related to failures in retrieving the shared survey point coordinates, such as missing survey point data or internal API errors.

## GetProjectPosition(Document, XYZ)

Retrieves the ProjectPosition for the specified origin point within the active project location.

```
public static ProjectPosition GetProjectPosition(Document document, XYZ origin)
```

## Parameters

`document` Document

The Autodesk.Revit.DB.Document containing the active project location. Must not be [null](#).

`origin` XYZ

The Autodesk.Revit.DB.XYZ point for which the project position is to be retrieved. Represents the origin in the project coordinate system.

## Returns

ProjectPosition

The ProjectPosition corresponding to the specified origin point, or [null](#) if the active project location is not set.

# Class MbRevitSpaceOldRepoUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Utility class for space operations and calculations using only Revit Space objects

```
public static class MbRevitSpaceOldRepoUtils
```

## Inheritance

[object](#) ← MbRevitSpaceOldRepoUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Fields

### ServiceCategories

```
public static readonly List<BuiltInCategory> ServiceCategories
```

## Field Value

[List](#)<BuiltInCategory>

## Methods

### AnalyzeCeilings(List<Ceiling>, RevitLinkInstance)

Determines the ceiling type for a space based on available ceilings

```
public static MbResult<CeilingAnalysisResult> AnalyzeCeilings(List<Ceiling> ceilings,  
RevitLinkInstance linkInstance)
```

## Parameters

`ceilings` [List](#)<Ceiling>

`linkInstance` RevitLinkInstance

Returns

[MbResult](#)<CeilingAnalysisResult>

## CalculatePolygonArea(List<XYZ>)

Calculates the area enclosed by a set of points using the shoelace formula

```
public static MbResult<double> CalculatePolygonArea(List<XYZ> boundaryPoints)
```

Parameters

`boundaryPoints` [List](#)<XYZ>

Returns

[MbResult](#)<double>

## CheckPointClashWithServices(XYZ, List<Element>, RevitLinkInstance, Document)

Checks for clashes between a point and service elements

```
public static MbResult<bool> CheckPointClashWithServices(XYZ gridPoint, List<Element> serviceElements, RevitLinkInstance linkInstance, Document document)
```

Parameters

`gridPoint` XYZ

`serviceElements` [List](#)<Element>

`linkInstance` RevitLinkInstance

`document` Document

Returns

[MbResult<bool>](#)

## CreateSpaceFromRoom(Room, RevitLinkInstance, Document)

Creates a space from a room with proper coordinate transformation

```
public static MbResult<Space> CreateSpaceFromRoom(Room room, RevitLinkInstance linkInstance, Document document)
```

Parameters

**room** Room

**linkInstance** RevitLinkInstance

**document** Document

Returns

[MbResult<Space>](#)

## DetermineCeilingHeight(Space, CeilingAnalysisResult, double, double)

Determines ceiling height for a space based on ceiling analysis

```
public static MbResult<double> DetermineCeilingHeight(Space space, CeilingAnalysisResult ceilingAnalysis, double minAboveFloorElevation = 0, double exposedSprinklerOffset = 0)
```

Parameters

**space** Space

**ceilingAnalysis** [CeilingAnalysisResult](#)

**minAboveFloorElevation** [double](#)

**exposedSprinklerOffset** [double](#)

Returns

[MbResult<double>](#)

## GetCeilingData(Ceiling, Document)

Gets ceiling data including type and face information

```
public static MbResult<CeilingData> GetCeilingData(Ceiling ceiling, Document linkDocument)
```

Parameters

**ceiling** Ceiling

**linkDocument** Document

Returns

[MbResult<CeilingData>](#)

## GetSpaceBoundaryPoints(Space)

Extracts boundary points from space boundary segments

```
public static MbResult<List<XYZ>> GetSpaceBoundaryPoints(Space space)
```

Parameters

**space** Space

Returns

[MbResult<List<XYZ>>](#)

## GetSpaceBoundarySegments(Space)

Gets boundary segments from a space with proper error handling

```
public static MbResult<List<List<Curve>>> GetSpaceBoundarySegments(Space space)
```

Parameters

**space** Space

Returns

[MbResult<List<List<Curve>>>](#)

## GetSpaceCenterPoint(Space)

Gets the center point of a space

```
public static MbResult<XYZ> GetSpaceCenterPoint(Space space)
```

Parameters

**space** Space

Returns

[MbResult<XYZ>](#)

## IsPointInPolygon(List<XYZ>, XYZ)

Checks if a point is inside a polygon using ray casting algorithm

```
public static MbResult<bool> IsPointInPolygon(List<XYZ> polygonPoints, XYZ testPoint)
```

Parameters

**polygonPoints** [List<XYZ>](#)

**testPoint** XYZ

Returns

[MbResult<bool>](#)

## IsPointInSpace(Space, XYZ)

Checks if a point is inside a space using polygon containment

```
public static MbResult<bool> IsPointInSpace(Space space, XYZ testPoint)
```

Parameters

**space** Space

**testPoint** XYZ

Returns

[MbResult<bool>](#)

## ProcessStructuralBeams(List<Tuple<Line, double, double>>, List<XYZ>, double)

Processes and filters structural beams for grid calculations

```
public static MbResult<BeamProcessingResult> ProcessStructuralBeams(List<Tuple<Line, double, double>> beamCenterLines, List<XYZ> spaceBoundaryPoints, double ceilingHeight)
```

Parameters

**beamCenterLines** [List<Tuple<Line, double, double>>](#)

**spaceBoundaryPoints** [List<XYZ>](#)

**ceilingHeight** [double](#)

Returns

[MbResult<BeamProcessingResult>](#)

## UpdateSpaceHeight(Space, double, Document)

Updates space height based on detected slab elevation

```
public static MbResult<bool> UpdateSpaceHeight(Space space, double aboveFloorElevation,  
Document document)
```

## Parameters

**space** Space

**aboveFloorElevation** [double](#)

**document** Document

## Returns

[MbResult<bool>](#)

# Class MbSelectionUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with family instances in Revit documents.

```
public class MbSelectionUtils
```

## Inheritance

[object](#) ← MbSelectionUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes methods for retrieving family instances based on specific criteria, such as visibility in a view and membership in specified built-in categories.

## Constructors

### MbSelectionUtils()

```
public MbSelectionUtils()
```

## Methods

### GetAllFamiliesByBuiltInCategories(IEnumerable<BuiltInCategory>, Document, ElementId)

Retrieves all family instances of the specified built-in categories that are visible in the given view.

```
public static MbResult<List<FamilyInstance>>
GetAllFamiliesByBuiltInCategories(IEnumerable<BuiltInCategory> builtInCategories, Document
document, ElementId viewId)
```

## Parameters

**builtInCategories** [IEnumerable](#)<BuiltInCategory>

The collection of BuiltInCategories to filter.

**document** Document

The Revit document.

**viewId** ElementId

The ID of the view to check visibility against.

## Returns

[MbResult](#)<[List](#)<FamilyInstance>>

An MbResult containing either a list of matching FamilyInstance elements or an error message.

# Class MbSpaceUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with Autodesk.Revit.DB.Mechanical.Space elements in Autodesk Revit.

```
public static class MbSpaceUtils
```

## Inheritance

[object](#) ← MbSpaceUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class contains static methods to assist in retrieving and manipulating Autodesk.Revit.DB.Mechanical.Space elements from the Revit model, particularly from the user's current selection in the active view.

## Methods

AnalyzeAboveCeilingGeometryAndGrid(Document, Space, RevitLinkInstance, View3D, double, bool, CeilingFaceGridParameters, double, bool)

Analyzes the geometry of ceilings above a given Autodesk.Revit.DB.Mechanical.Space and generates a grid for sprinkler placement.

```
public static MbResult<(List<Ceiling> Ceilings, Ceiling MajorCeiling, MbCeilingType MbCeilingType, Face MajorFace, XYZ PointInCeiling, double CeilingHeight, CeilingFaceGrid CeilingGrid)> AnalyzeAboveCeilingGeometryAndGrid(Document doc, Space space, RevitLinkInstance ceilingsLinkInstance, View3D analysisView, double sprinklerOffset = 0, bool drawCurvesInRevit = true, CeilingFaceGridParameters gridParameters = null, double ceilingElementsDetectionRange = 6, bool sharedPositioningEnabled = false)
```

## Parameters

**doc** Document

The active Revit Autodesk.Revit.DB.Document. Must not be **null**.

**space** Space

The Autodesk.Revit.DB.Mechanical.Space to analyze. Must not be **null**.

**ceilingsLinkInstance** RevitLinkInstance

The Autodesk.Revit.DB.RevitLinkInstance containing ceiling elements. Must not be **null**.

**analysisView** View3D

The Autodesk.Revit.DB.View3D used for geometric analysis. Must not be **null**.

**sprinklerOffset** [double](#)

Optional offset to subtract from ceiling height for sprinkler placement.

**drawCurvesInRevit** [bool](#)

If **true**, draws boundary curves in Revit for debugging.

**gridParameters** [CeilingFaceGridParameters](#)

Optional [CeilingFaceGridParameters](#) for grid configuration. If **null**, defaults are used.

**ceilingElementsDetectionRange** [double](#)

Detection range (in feet) for ceiling elements above the space. Must be greater than zero.

**sharedPositioningEnabled** [bool](#)

## Returns

[MbResult](#)<[List](#)<Ceiling> [Ceilings](#), Ceiling [MajorCeiling](#), [MbCeilingType](#) [MbCeilingType](#), Face [MajorFace](#), XYZ [PointInCeiling](#), [double](#) [CeilingHeight](#), [CeilingFaceGrid](#) [CeilingGrid](#)>>

An [MbResult](#) containing a tuple with:

- [List](#): All detected ceilings above the space.
- Ceiling: The major ceiling used for grid generation.
- [MbCeilingType](#): The type of the major ceiling (e.g., TiledMajor, PlasterBoard, Exposed).
- Face: The major face of the ceiling.

- Autodesk.Revit.DB.XYZ: A representative point in the ceiling face.
- [double](#): The calculated ceiling height (Z coordinate).
- [CeilingFaceGrid](#): The generated grid for sprinkler placement.

If the operation fails, the result contains an error message describing the reason. Possible errors include:

- "Document is null."
- "Space is null."
- "Ceilings link instance is null."
- "Analysis view (View3D) is null."
- Errors from [GetChainedLoops\(Document, List<List<Curve>>, bool, double\)](#) (e.g., invalid boundary geometry).
- Errors from [GetAboveCeilingOfSpace\(XYZ, Document, RevitLinkInstance, List<Curve>, Transform, List<BuiltInCategory>, double, bool\)](#) (e.g., failed solid creation, no ceilings found).
- Errors from MBF.Revit.Data.Utils.MbSpaceUtils.DetermineCeilingType(System.Collections.Generic.List{Autodesk.Revit.DB.Ceiling},Autodesk.Revit.DB.RevitLinkInstance) (e.g., link document is null, no valid ceiling face).
- Errors from MBF.Revit.Data.Utils.MbSpaceUtils.GetMinAboveFloorElevation(Autodesk.Revit.DB.XYZ, Autodesk.Revit.DB.RevitLinkInstance,Autodesk.Revit.DB.Document,Autodesk.Revit.DB.View3D) (e.g., failed raycast, transformation errors).
- Errors from [GetSpaceBoundaryPoints\(Space\)](#) (e.g., failed to extract boundary points).
- Errors from [Create\(Ceiling, Document, RevitLinkInstance, Space, CeilingFaceGridParameters, double, List<XYZ>, double\)](#) (e.g., grid generation failure).

## ClosestDistanceToFramingElement(Document, View3D, XYZ)

```
public static MbResult<(double Distance, XYZ IntersectionPoint)>
ClosestDistanceToFramingElement(Document doc, View3D view3D, XYZ originPoint)
```

### Parameters

**doc** Document

**view3D** View3D

**originPoint** XYZ

### Returns

[MbResult<\(double Distance, XYZ IntersectionPoint\)>](#)

## GetAboveCeilingOfSpace(XYZ, Document, RevitLinkInstance, List<Curve>, Transform, List<BuiltInCategory>, double, bool)

Retrieves a list of ceilings located above a specified point within a given range in a Revit model.

```
public static MbResult<List<Ceiling>> GetAboveCeilingOfSpace(XYZ point, Document parentDocument, RevitLinkInstance linkInstance, List<Curve> segments, Transform originalElementTransform, List<BuiltInCategory> builtinCategories, double ceilingElementsRange, bool sharedPositioningEnabled)
```

### Parameters

**point** XYZ

The point in space from which to search for ceilings. Cannot be [null](#).

**parentDocument** Document

The parent Revit document containing the space. Cannot be [null](#).

**linkInstance** RevitLinkInstance

The Revit link instance to search within. Cannot be [null](#).

**segments** [List](#)<Curve>

A list of curves representing the boundaries of the space. Must contain at least one curve.

**originalElementTransform** Transform

The transform of the original element defining the space. Cannot be [null](#).

**builtinCategories** [List](#)<BuiltInCategory>

A list of built-in categories to filter ceiling elements. Must contain at least one category.

**ceilingElementsRange** [double](#)

The maximum range, in Revit units, to search for ceiling elements above the point. Must be greater than zero.

**sharedPositioningEnabled** [bool](#)

A value indicating whether shared positioning is enabled for the link instance.

## Returns

[MbResult<List<Ceiling>>](#)

An [MbResult<T>](#) containing a list of Ceiling objects if the operation succeeds, or an error message if it fails.

## Remarks

This method identifies ceilings above the specified point by analyzing the geometry of the space and applying filters to detect intersecting ceiling elements. It supports both linked Revit models and shared positioning configurations.

## GetCeilingsFromLinkByView(RevitLinkInstance, ElementIntersectsSolidFilter)

Retrieves a list of ceilings from a linked Revit document that intersect with the specified solid filter.

```
public static MbResult<List<Ceiling>> GetCeilingsFromLinkByView(RevitLinkInstance  
linkInstance, ElementIntersectsSolidFilter solidFilter)
```

## Parameters

**linkInstance** RevitLinkInstance

The Autodesk.Revit.DB.RevitLinkInstance representing the linked Revit document. Must not be [null](#).

**solidFilter** ElementIntersectsSolidFilter

The Autodesk.Revit.DB.ElementIntersectsSolidFilter used to filter ceilings based on intersection criteria. Must not be [null](#).

## Returns

[MbResult<List<Ceiling>>](#)

An [MbResult<T>](#) containing a list of Ceiling elements that match the filter criteria. If the operation fails, the result contains an error message describing the failure.

## Remarks

This method uses the FilteredElementCollector to retrieve ceilings from the linked document. Ensure that the linked document is loaded and accessible before calling this method.

## GetSolidInLinkFromSpace(List<Curve>, Transform, Document, RevitLinkInstance, double, bool)

```
public static MbResult<Solid> GetSolidInLinkFromSpace(List<Curve> segments, Transform  
originalElementTransform, Document parentDoc, RevitLinkInstance linkInstance, double  
detectionRange, bool sharedPositioningEnabled)
```

### Parameters

**segments** [List](#)<Curve>

**originalElementTransform** Transform

**parentDoc** Document

**linkInstance** RevitLinkInstance

**detectionRange** [double](#)

**sharedPositioningEnabled** [bool](#)

### Returns

[MbResult](#)<Solid>

## GetSpacesFromSelection(UIDocument)

Retrieves a list of Autodesk.Revit.DB.Mechanical.Space elements from the current selection in the Revit UI.

```
public static List<Space> GetSpacesFromSelection(UIDocument uiDoc)
```

### Parameters

**uiDoc** UIDocument

The UIDocument representing the active Revit document and user interface context.

## Returns

[List](#) <Space>

A list of Autodesk.Revit.DB.Mechanical.Space elements selected by the user. Returns an empty list if no valid Autodesk.Revit.DB.Mechanical.Space elements are selected.

## Remarks

This method filters the current selection to include only valid Autodesk.Revit.DB.Mechanical.Space elements. Invalid or non-Autodesk.Revit.DB.Mechanical.Space elements in the selection are ignored.

# Class MbSprinklersUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with sprinkler elements in Revit.

```
public static class MbSprinklersUtils
```

## Inheritance

[object](#) ← MbSprinklersUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class contains methods to assist in identifying and working with Revit elements categorized as sprinklers. It is designed to simplify common operations related to sprinkler elements, such as filtering and casting.

## Methods

### GetSprinklerInstances(IEnumerable<Element>)

Retrieves all FamilyInstance elements categorized as sprinklers from the provided collection of Revit elements.

```
public static IEnumerable<FamilyInstance> GetSprinklerInstances(IEnumerable<Element>  
selectedElements)
```

#### Parameters

**selectedElements** [IEnumerable](#)<Element>

A collection of Element objects to search within.

#### Returns

## [IEnumerable](#) <FamilyInstance>

An [IEnumerable<T>](#) containing all elements from `selectedElements` that are categorized as `BuiltInCategory.OST_Sprinklers`.

## Remarks

This method filters the input collection to include only elements whose category is "Sprinklers", then casts those elements to FamilyInstance for further use. The filtering logic uses conditional compilation to support different Revit versions.

# Class MbSurveyPointUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with survey points in Revit documents.

```
public static class MbSurveyPointUtils
```

## Inheritance

[object](#) ← MbSurveyPointUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes methods to retrieve and process shared survey point data from Revit documents. It is designed to simplify access to survey point information, which is commonly used in site coordination and geolocation workflows.

## Methods

### GetSharedSurveyPointCoordinates(Document)

Retrieves the coordinates of the shared survey point from the specified Revit document.

```
public static MbResult<XYZ> GetSharedSurveyPointCoordinates(Document document)
```

#### Parameters

**document** Document

The Autodesk.Revit.DB.Document instance representing the Revit model from which to retrieve the shared survey point coordinates.

#### Returns

## [MbResult<XYZ>](#)

Returns an [MbResult<T>](#) containing the coordinates of the shared survey point as an Autodesk.Revit.DB.XYZ object if successful. If no shared survey point is found or an error occurs, returns a failed [MbResult<T>](#) with an appropriate error message.

## Remarks

This method searches for elements of category `BuiltInCategory.OST_SharedBasePoint` and type `BasePoint` that are marked as shared within the provided document. It extracts the East-West, North-South, and Elevation parameters from the first shared survey point found and returns them as an Autodesk.Revit.DB.XYZ coordinate.

# Class MbUnitUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with measurement units in Revit.

```
public static class MbUnitUtils
```

## Inheritance

[object](#) ← MbUnitUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes methods to retrieve and standardize unit symbols for length measurements in the active Revit document. It supports both modern and legacy Revit versions by handling differences in unit type representations.

## Methods

### GetLengthUnitSymbol()

Retrieves the unit symbol used for length in the active Revit document.

```
public static string GetLengthUnitSymbol()
```

Returns

[string](#)

A string representing the standardized symbol of the length unit (e.g., "mm", "ft", "in", etc.). If the unit type cannot be identified, returns "Unknown".

### GetStandardUnitSymbol(object)

Maps Revit unit types to standardized short symbols.

```
public static string GetStandardUnitSymbol(object unitTypeId)
```

## Parameters

**unitTypeId** [object](#)

The unit type identifier, either a ForgeTypeid or DisplayUnitType, depending on Revit version.

## Returns

[string](#)

A short string symbol representing the unit (e.g., "mm", "cm", "ft-in"). Returns a user-readable label from Revit's Autodesk.Revit.DB.LabelUtils as a fallback. Returns "Unknown" if mapping cannot be resolved.

## ToInternalUnits(double)

Converts millimeters to Revit internal units (feet)

```
public static double ToInternalUnits(this double millimeters)
```

## Parameters

**millimeters** [double](#)

Value in millimeters

## Returns

[double](#)

Value in Revit internal units

# Class MbViewUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utility methods for working with Revit views, including graphical view checks, zooming to elements, creating temporary 3D views, and managing view isolation.

```
public static class MbViewUtils
```

## Inheritance

[object](#) ← MbViewUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class contains static methods designed to simplify common operations related to Revit views. It includes functionality for determining view types, manipulating graphical views, and creating or managing temporary 3D views. These utilities are intended to streamline workflows in Revit add-ins and enhance user interaction with views.

## Methods

### Create3DView(Document)

Creates a new isometric 3D view in the specified document.

```
public static MbResult<View3D> Create3DView(Document doc)
```

#### Parameters

**doc** Document

The Revit document in which the 3D view will be created. Cannot be null.

#### Returns

## MbResult<View3D>

MbResult<T> containing the result of the operation:

- **Success:** Returns a successful MbResult<T> containing the newly created Autodesk.Revit.DB.View3D.
- **Failure:** Returns a failed MbResult<T> with an error message if:
  - No 3D view type is found to copy from.
  - The Autodesk.Revit.DB.ViewFamilyType could not be retrieved.
  - Failure to create the 3D view.

## Remarks

This method attempts to create a new isometric 3D view by copying the settings from an existing non-template 3D view in the document. The new view is named "Temp3DMFire" and has its detail level set to ViewDetailLevel.Fine.

## EnsureTemp3DViewIsOpen(Document)

Creates and opens the temporary 3D view if it's not already opened in the UI.

```
public static void EnsureTemp3DViewIsOpen(Document activeDoc)
```

## Parameters

**activeDoc** Document

The active document wrapper.

## GetAllPlanViews(Document)

Retrieves all non-template plan views from the specified Revit document.

```
public static IList<ViewPlan> GetAllPlanViews(Document doc)
```

## Parameters

**doc** Document

The Revit document from which to retrieve the plan views. Cannot be null.

## Returns

[IList](#) <ViewPlan>

A list of Autodesk.Revit.DB.ViewPlan objects representing all non-template plan views in the document. Returns an empty list if no such views are found.

## GetAllThreeDViews(Document)

Retrieves all 3D views in the specified Revit document that are not templates.

```
public static List<View3D> GetAllThreeDViews(Document doc)
```

## Parameters

**doc** Document

The Revit document from which to retrieve the 3D views. Cannot be [null](#).

## Returns

[List](#) <View3D>

A list of Autodesk.Revit.DB.View3D objects representing all non-template 3D views in the document. Returns an empty list if no such views are found.

## GetOrCreateTemp3D(Document)

Retrieves an existing 3D view named "Temp3DMBFire" from the specified document, or creates a new one if it does not exist.

```
public static MbResult<View3D> GetOrCreateTemp3D(Document doc)
```

## Parameters

**doc** Document

The Autodesk.Revit.DB.Document from which to retrieve or create the 3D view. This parameter cannot be [null](#).

## Returns

[MbResult<View3D>](#)

A [MbResult<T>](#) containing the 3D view. If a view named "Temp3DMFire" exists, it is returned; otherwise, a new 3D view is created and returned.

## Remarks

This method first searches for an existing 3D view with the name "Temp3DMFire". If found, it returns the existing view wrapped in a successful [MbResult<T>](#). If no such view exists, a new 3D view is created and returned.

## GetTemp3DNoTransaction(Document)

Retrieves a temporary 3D view for the specified document without initiating a transaction.

```
public static MbResult<View3D> GetTemp3DNoTransaction(Document doc)
```

## Parameters

**doc** Document

The document for which the temporary 3D view is to be retrieved or created. Cannot be null.

## Returns

[MbResult<View3D>](#)

A [MbResult<T>](#) containing the temporary 3D view if successful, or an error result if the operation fails.

## IsGraphicalView(View)

Determines whether the specified Autodesk.Revit.DB.View represents a graphical view.

```
public static bool IsGraphicalView(View view)
```

## Parameters

**view** View

The Autodesk.Revit.DB.View to evaluate. Must not be [null](#).

Returns

[bool](#)

[true](#) if the [view](#) is a graphical view, such as a floor plan, section, or 3D view; otherwise, [false](#).

Remarks

Graphical views include types such as floor plans, ceiling plans, sections, elevations, 3D views, drafting views, area plans, engineering plans, details, and legends. Non-graphical views include schedules, reports, and undefined views.

## ResetTemporaryIsolate()

Resets the temporary isolate mode for the active view, if it is currently enabled.

```
public static void ResetTemporaryIsolate()
```

Remarks

This method checks if the active view has temporary isolate mode enabled and, if so, disables it. If no active view exists or temporary isolate mode is not active, the method performs no action.

## TemporarilyIsolateElement(Element)

Temporarily isolates the specified element in the active graphical view.

```
public static bool TemporarilyIsolateElement(Element element)
```

Parameters

[element](#) Element

The element to isolate. Must belong to the document of the active view.

Returns

[bool](#)

[true](#) if the element was successfully isolated; otherwise, [false](#).

## Remarks

This method isolates the specified element temporarily in the active view of the document. Isolation is only supported in graphical views. If the active view is not graphical, the method displays a warning message and returns [false](#). If an error occurs during the isolation process, an error message is displayed, and the method returns [false](#).

## ZoomToElement(Element)

Zooms and centers the active view on the specified element's bounding box.

```
public static void ZoomToElement(Element element)
```

## Parameters

**element** Element

The element to zoom to. Must belong to the active document and have a valid bounding box in the active view.

## Remarks

This method adjusts the active view to focus on the bounding box of the specified element. If the bounding box cannot be retrieved or the active view is not a graphical view (e.g., plan, 3D view, section, or legend), an error message is displayed, and no action is taken. The method also optionally selects the element in the active view.

## ZoomToElement(ElementId)

Zooms and centers the active view on the specified element by its ElementId.

```
public static void ZoomToElement(ElementId elementId)
```

## Parameters

**elementId** ElementId

The ElementId of the element to zoom to. Must belong to the active document.

## Remarks

This method adjusts the active view to focus on the bounding box of the specified element. If the bounding box cannot be retrieved, an error message is displayed. The method works only in graphical views such as plan, 3D, section, or legend views. If the active view is not a supported graphical view, an error message is displayed.

## ZoomToElement(FamilyInstance)

Zooms and centers the active view on the specified FamilyInstance element.

```
public static void ZoomToElement(FamilyInstance instance)
```

## Parameters

**instance** FamilyInstance

The FamilyInstance to zoom to. Must belong to the active document.

## Remarks

This method adjusts the active view to focus on the bounding box of the specified element. If the bounding box cannot be retrieved, an error message is displayed. The method works only in graphical views such as plan, 3D, section, or legend views. If the active view is not a supported graphical view, an error message is displayed.

# Enum PipePointType

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Represents a specific point along a pipe, either at the start or the end.

```
public enum PipePointType
```

## Fields

**End = 1**

Refers to the end point of the pipe geometry. Equivalent to `pipe.get_EndPoint(1)`.

**Start = 0**

Refers to the start point of the pipe geometry. Equivalent to `pipe.get_EndPoint(0)`.

# Class RevitUnitUtils

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Determines whether the units of the specified Revit document are metric.

```
public static class RevitUnitUtils
```

## Inheritance

[object](#) ← RevitUnitUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This method checks the length unit settings of the provided Revit document to determine if they are configured to use metric units. Metric units include meters, centimeters, and millimeters.

## Methods

### IsMetric(Document)

Determines whether the specified document uses metric units for length measurements.

```
public static bool IsMetric(this Document doc)
```

#### Parameters

**doc** Document

The document to evaluate.

#### Returns

[bool](#)

[true](#) if the document uses metric units such as meters, centimeters, or millimeters; otherwise, [false](#).

# Class XyzExtensibleStorage

Namespace: [MBF.Revit.Data.Utils](#)

Assembly: MBF.Revit.Data.dll

Provides utilities for reading and writing Autodesk.Revit.DB.XYZ start point data for Autodesk.Revit.DB.Plumbing.Pipe elements using Revit's Extensible Storage API.

```
public static class XyzExtensibleStorage
```

## Inheritance

[object](#) ← XyzExtensibleStorage

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## GetEntityValue(Entity, Field)

Retrieves the value of the specified field from the given entity as an Autodesk.Revit.DB.XYZ object.

```
public static XYZ GetEntityValue(Entity entity, Field field)
```

### Parameters

**entity** Entity

The Autodesk.Revit.DB.ExtensibleStorage.Entity from which the field value is retrieved. Cannot be [null](#).

**field** Field

The Autodesk.Revit.DB.ExtensibleStorage.Field that specifies the data to retrieve. Cannot be [null](#).

### Returns

XYZ

An Autodesk.Revit.DB.XYZ object representing the value of the specified field in the entity.

## Remarks

The method retrieves the value using the appropriate unit system based on the Revit version. For Revit 2021 or later, the method uses a unit system derived from the Autodesk.Revit.DB.SpecTypeId.Length specification. For earlier versions, it defaults to DisplayUnitType.DUT\_FEET\_FRACTIONAL\_INCHES.

## GetNearestEndPoints(Pipe)

Retrieves the endpoints of the specified pipe's location curve.

```
public static List<XYZ> GetNearestEndPoints(this Pipe pipe)
```

### Parameters

**pipe** Pipe

The pipe whose endpoints are to be retrieved. Must have a valid location curve.

### Returns

[List](#)<XYZ>

A list of Autodesk.Revit.DB.XYZ objects representing the start and end points of the pipe's location curve. If the pipe does not have a location curve, an empty list is returned.

## Remarks

This method assumes the pipe's location is a Autodesk.Revit.DB.LocationCurve. If the location is not a Autodesk.Revit.DB.LocationCurve, the method will return an empty list without throwing an exception.

## GetNearestEndPoints(Pipe, XYZ)

Returns the pipe's endpoints sorted by proximity to a given point.

```
public static List<XYZ> GetNearestEndPoints(this Pipe p, XYZ referencePoint)
```

### Parameters

**p** Pipe

The pipe.

**referencePoint XYZ**

The point to compare against.

Returns

[List](#) <XYZ>

List of endpoints with the nearest one first.

## ReadStartPoint(Pipe)

Retrieves the stored start point of the given Autodesk.Revit.DB.Plumbing.Pipe, if available and valid.

```
public static XYZ ReadStartPoint(Pipe pipe)
```

Parameters

**pipe** Pipe

The pipe from which to retrieve the stored Autodesk.Revit.DB.XYZ start point.

Returns

XYZ

The nearest endpoint to the stored start point if valid and close to the actual pipe ends; otherwise, **null**.

## SetStartPoint(Pipe, XYZ, string)

Sets the start point of a pipe in extensible storage, choosing the nearest pipe endpoint. Skips overwriting if already set by another tool, unless the caller is explicitly allowed.

```
public static void SetStartPoint(Pipe pipe, XYZ startPoint, string caller = null)
```

Parameters

**pipe** Pipe

The pipe to set the start point for.

**startPoint** XYZ

The picked point by the user.

**caller** [string ↗](#)

The name of the calling method, used to control write permissions.

# Namespace MBF.Revit.Geometry.Extensions

## Classes

### [BoundingBoxExtensions](#)

Provides extension methods for working with Autodesk.Revit.DB.BoundingBoxXYZ objects.

### [BoundingBoxVisualizerExtensions](#)

Provides extension methods for visualizing bounding boxes as 2D outlines in a Revit document.

### [ClipperConversionExtensions](#)

Provides extension methods for converting CurveLoop and Autodesk.Revit.DB.Curve objects into Clipper-compatible path representations.

### [ClipperExtensions](#)

Provides extension methods for working with Clipper2Lib.PathsD and Clipper2Lib.PathD collections, as well as converting Clipper2Lib.PathD objects to Revit CurveLoop instances.

### [CurveLoopExtensions](#)

Provides extension methods for converting CurveLoop objects into Clipper-compatible path representations, such as Clipper2Lib.PathD and Clipper2Lib.PathsD.

### [CurveLoopVisualizerExtensions](#)

Provides extension methods for visualizing and creating Revit elements from CurveLoop objects.

### [CurveVisualizerExtensions](#)

Provides extension methods for visualizing Autodesk.Revit.DB.Curve instances as 2D geometry in Revit, either using DirectShape elements in model views or Autodesk.Revit.DB.DetailCurve elements in drafting views.

### [DirectShapeVisualization](#)

Provides methods for creating and managing DirectShape elements in a Revit document.

### [PathDVisualizerExtensions](#)

Provides extension methods for visualizing Clipper2Lib.PathD and Clipper2Lib.PathsD objects as 2D outlines in a Revit document using DirectShape elements.

### [XyzVisualizationsExtensions](#)

Provides extension methods for visualizing Autodesk.Revit.DB.XYZ points as 3D objects in a Revit document.

# Class BoundingBoxExtensions

Namespace: [MBF.Revit.Geometry.Extensions](#)

Assembly: MBF.Revit.Geometry.dll

Provides extension methods for working with Autodesk.Revit.DB.BoundingBoxXYZ objects.

```
public static class BoundingBoxExtensions
```

## Inheritance

[object](#) ← BoundingBoxExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes methods to convert Autodesk.Revit.DB.BoundingBoxXYZ instances into other geometric representations, such as CurveLoop objects, for use in various geometric and modeling operations.

## Methods

### ToCurveLoop(BoundingBoxXYZ)

Converts a Autodesk.Revit.DB.BoundingBoxXYZ into a closed rectangular CurveLoop in the XY plane.

```
public static MbResult<CurveLoop> ToCurveLoop(this BoundingBoxXYZ boundingBox)
```

#### Parameters

**boundingBox** BoundingBoxXYZ

The bounding box to convert.

#### Returns

[MbResult](#)<CurveLoop>

An [MbResult<T>](#):

- **Success:** Contains the rectangular CurveLoop.
- **Failure:** "BoundingBoxXYZ is null."
- **Failure:** "Failed to convert BoundingBoxXYZ to CurveLoop: [exception message]"

# Class BoundingBoxVisualizerExtensions

Namespace: [MBF.Revit.Geometry.Extensions](#)

Assembly: MBF.Revit.Geometry.dll

Provides extension methods for visualizing bounding boxes as 2D outlines in a Revit document.

```
public static class BoundingBoxVisualizerExtensions
```

## Inheritance

[object](#) ← BoundingBoxVisualizerExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes methods to visualize single or multiple bounding boxes as 2D shapes in the specified Revit document. The visualizations are created as red outlines by default, but an optional color can be specified.

## Methods

### VisualizeBoundingBox2D(BoundingBoxXYZ, Document, Color)

Visualizes a 2D representation of the specified Autodesk.Revit.DB.BoundingBoxXYZ in the given Revit document.

```
public static void VisualizeBoundingBox2D(this BoundingBoxXYZ boundingBox, Document doc,  
Color color = null)
```

#### Parameters

**boundingBox** BoundingBoxXYZ

The 2D bounding box to visualize. Cannot be [null](#).

**doc** Document

The Revit document where the bounding box will be visualized. Cannot be [null](#).

#### **color** Color

The color to use for the visualization. If [null](#), the default color is red (RGB: 255, 0, 0).

## Remarks

This method creates a 2D visualization of the bounding box as a direct shape in the specified document. The visualization is created within a transaction, which is started and committed by this method.

## VisualizeBoundingBoxes2D(IEnumerable<BoundingBoxXYZ>, Document, Color)

Visualizes a collection of 2D bounding boxes in the specified Revit document.

```
public static void VisualizeBoundingBoxes2D(this IEnumerable<BoundingBoxXYZ> boundingBoxes, Document doc, Color color = null)
```

## Parameters

### **boundingBoxes** [IEnumerable](#)<BoundingBoxXYZ>

A collection of Autodesk.Revit.DB.BoundingBoxXYZ objects representing the 2D bounding boxes to visualize.

### **doc** Document

The Autodesk.Revit.DB.Document in which the bounding boxes will be visualized. This cannot be [null](#).

#### **color** Color

An optional Autodesk.Revit.DB.Color to use for the bounding box visualization. If not specified, the default color is red (RGB: 255, 0, 0).

## Remarks

This method creates 2D visual representations of the provided bounding boxes as direct shapes in the Revit document. If a bounding box in the collection is [null](#), it will be skipped.

# Class ClipperConversionExtensions

Namespace: [MBF.Revit.Geometry.Extensions](#)

Assembly: MBF.Revit.Geometry.dll

Provides extension methods for converting CurveLoop and Autodesk.Revit.DB.Curve objects into Clipper-compatible path representations.

```
public static class ClipperConversionExtensions
```

## Inheritance

[object](#) ← ClipperConversionExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This static class includes methods to convert individual CurveLoop objects, collections of CurveLoop objects, and lists of Autodesk.Revit.DB.Curve objects into Clipper2Lib.PathD and Clipper2Lib.PathsD formats, which are compatible with the Clipper library. The methods handle various conversion scenarios, including optional deduplication of points and ensuring path closure.

## Methods

### ToClipperPathD(CurveLoop)

Converts a CurveLoop into a Clipper-compatible Clipper2Lib.PathD by collecting its curve endpoints.

```
public static MbResult<PathD> ToClipperPathD(this CurveLoop curveLoop)
```

#### Parameters

**curveLoop** CurveLoop

The CurveLoop to convert.

#### Returns

## [MbResult<PathD>](#)

An [MbResult<T>](#):

- **Success:** Contains the converted Clipper2Lib.PathD.
- **Failure:** "CurveLoop is null."
- **Failure:** "The CurveLoop must have at least 3 points to form a polygon."
- **Failure:** "Failed to convert CurveLoop to PathD: [exception message]"

## ToClipperPathD(List<Curve>, bool, bool)

Converts a list of Autodesk.Revit.DB.Curve objects into a Clipper2Lib.PathD object.

```
public static MbResult<PathD> ToClipperPathD(this List<Curve> curves, bool ensureClosed = true, bool removeDuplicates = true)
```

### Parameters

**curves** [List<Curve>](#)

The list of Autodesk.Revit.DB.Curve objects to convert.

**ensureClosed** [bool](#)

A boolean value indicating whether the resulting path should be closed. If [true](#), ensures the path is closed by connecting the last point to the first.

**removeDuplicates** [bool](#)

A boolean value indicating whether duplicate points should be removed from the path. If [true](#), duplicate points are removed.

### Returns

[MbResult<PathD>](#)

An [MbResult<T>](#) containing the resulting Clipper2Lib.PathD if successful; otherwise, an error message indicating the failure reason.

### Remarks

This method validates the input curves and processes them to create a Clipper2Lib.PathD. It handles optional deduplication of points and ensures the path is closed if specified.

## ToClipperPathsD(IEnumerable<CurveLoop>)

Converts a collection of CurveLoop objects into a Clipper-compatible Clipper2Lib.PathsD.

```
public static MbResult<PathsD> ToClipperPathsD(this IEnumerable<CurveLoop> curveLoops)
```

### Parameters

`curveLoops` [IEnumerable](#)<CurveLoop>

The CurveLoop collection to convert.

### Returns

[MbResult](#)<PathsD>

An [MbResult](#)<T>:

- **Success:** Contains the converted Clipper2Lib.PathsD.
- **Failure:** "CurveLoop collection is null."
- **Failure:** Any error from [ToClipperPathD\(CurveLoop\)](#) such as "CurveLoop is null." or invalid geometry.
- **Failure:** "Failed to convert CurveLoops to PathsD: [exception message]"

## ToClipperPathsD(IEnumerable<List<Curve>>, bool, bool)

Converts a collection of curve collections into Clipper paths represented by Clipper2Lib.PathsD.

```
public static MbResult<PathsD> ToClipperPathsD(this IEnumerable<List<Curve>>
curveCollections, bool ensureClosed = true, bool removeDuplicates = true)
```

### Parameters

`curveCollections` [IEnumerable](#)<[List](#)<Curve>>

The collection of curve collections to convert. Cannot be [null](#).

`ensureClosed` [bool](#)

A value indicating whether to ensure that each path is closed. If [true](#), paths will be closed if they are not already.

`removeDuplicates` [bool](#)

A value indicating whether to remove duplicate points from the paths. If [true](#), duplicate points will be removed.

## Returns

### [MbResult](#)<PathsD>

An [MbResult](#)<T> containing the converted Clipper2Lib.PathsD if successful; otherwise, an error message indicating the failure reason.

## Remarks

Each curve collection is converted individually, and only paths with a sufficient number of points are included in the result. The method handles exceptions internally and returns a failure result with an error message if an exception occurs.

## ToClipperPoints(Curve, bool)

Converts a Autodesk.Revit.DB.Curve to a list of Clipper2Lib.PointD objects.

```
public static MbResult<List<PointD>> ToClipperPoints(this Curve curve, bool includeEndPoint = false)
```

## Parameters

### curve Curve

The curve to convert. Cannot be null.

### includeEndPoint [bool](#)

A boolean value indicating whether to include the end point of the curve in the result. If [true](#), the end point is included; otherwise, it is not.

## Returns

### [MbResult](#)<[List](#)<PointD>>

An [MbResult](#)<T> containing a list of Clipper2Lib.PointD objects representing the curve. Returns a failure result if the conversion fails or if the curve is null.

## Remarks

The method always includes the start point of the curve in the result. If `includeEndPoint` is set to `true`, the end point is also included. The method handles any conversion errors and returns a failure result with an appropriate error message.

## ToPathsD(CurveLoop)

Wraps a single CurveLoop into a Clipper2Lib.PathsD collection.

```
public static MbResult<PathsD> ToPathsD(this CurveLoop curveLoop)
```

### Parameters

`curveLoop` CurveLoop

The CurveLoop to wrap.

### Returns

[MbResult<PathsD>](#)

An [MbResult<T>](#):

- **Success:** Contains a Clipper2Lib.PathsD with one converted path.
- **Failure:** Any error from [ToClipperPathD\(CurveLoop\)](#), such as null or invalid geometry.

# Class ClipperExtensions

Namespace: [MBF.Revit.Geometry.Extensions](#)

Assembly: MBF.Revit.Geometry.dll

Provides extension methods for working with Clipper2Lib.PathsD and Clipper2Lib.PathD collections, as well as converting Clipper2Lib.PathD objects to Revit CurveLoop instances.

```
public static class ClipperExtensions
```

## Inheritance

[object](#) ← ClipperExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes utility methods for manipulating Clipper paths and integrating them with Revit geometry.

## Methods

### AddPathDToPathsD(PathsD, PathD)

Adds a single Clipper2Lib.PathD to an existing Clipper2Lib.PathsD collection.

```
public static PathsD AddPathDToPathsD(this PathsD pathsD, PathD pathD)
```

#### Parameters

**pathsD** PathsD

The Clipper2Lib.PathsD collection to which the path will be added.

**pathD** PathD

The Clipper2Lib.PathD to add.

Returns

PathsD

The updated Clipper2Lib.PathsD collection containing the added path.

## ToCurveLoop(PathD)

Converts a Clipper2Lib.PathD into a Revit CurveLoop by connecting its points with lines.

```
public static MbResult<CurveLoop> ToCurveLoop(this PathD pathD)
```

Parameters

**pathD** PathD

The Clipper path to convert.

Returns

[MbResult](#)<CurveLoop>

An [MbResult](#)<T>:

- **Success:** Contains the resulting CurveLoop.
- **Failure:** "PathD is null."
- **Failure:** "PathD must contain at least two points to create a CurveLoop."
- **Failure:** "Failed to convert PathD to CurveLoop: [exception message]"

# Class CurveLoopExtensions

Namespace: [MBF.Revit.Geometry.Extensions](#)

Assembly: MBF.Revit.Geometry.dll

Provides extension methods for converting CurveLoop objects into Clipper-compatible path representations, such as Clipper2Lib.PathD and Clipper2Lib.PathsD.

```
public static class CurveLoopExtensions
```

## Inheritance

[object](#) ← CurveLoopExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

These methods facilitate the conversion of geometric curve data into formats compatible with the Clipper library, enabling operations such as polygon clipping and path manipulation. The methods handle common validation scenarios, such as ensuring the input is not null and that the geometry meets the minimum requirements for conversion.

# Class CurveLoopVisualizerExtensions

Namespace: [MBF.Revit.Geometry.Extensions](#)

Assembly: MBF.Revit.Geometry.dll

Provides extension methods for visualizing and creating Revit elements from CurveLoop objects.

```
public static class CurveLoopVisualizerExtensions
```

## Inheritance

[object](#) ← CurveLoopVisualizerExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes methods for creating 2D visualizations of CurveLoop objects using DirectShape elements and for generating Autodesk.Revit.DB.DetailCurve elements in drafting views. These methods are designed to work with the active view of a Revit document and provide optional customization, such as specifying outline colors.

## Methods

### CreateCurveLoopWithDetailCurves(CurveLoop, Document, ViewDrafting, Color)

Creates Autodesk.Revit.DB.DetailCurve elements in a Autodesk.Revit.DB.ViewDrafting to represent the given CurveLoop.

```
public static MbResult<List<DetailCurve>> CreateCurveLoopWithDetailCurves(this CurveLoop  
curveLoop, Document doc, ViewDrafting draftingView, Color color = null)
```

#### Parameters

**curveLoop** CurveLoop

The CurveLoop to draw.

**doc** Document

The current Revit document.

**draftingView** ViewDrafting

The drafting view where detail curves will be placed.

**color** Color

Optional line color; defaults to red if not provided.

Returns

[MbResult<List<DetailCurve>>](#)

An [MbResult<T>](#) where **T** is [List<T>](#):

- **Success:** Contains the created list of Autodesk.Revit.DB.DetailCurve elements.
- **Failure:** "CurveLoop is null."
- **Failure:** "Document or drafting view is null."
- **Failure:** "Failed to create detail curves: [exception message]"

## VisualizeCurveLoop2D(CurveLoop, Document, Color)

Visualizes a single CurveLoop as a 2D outline using a DirectShape element in the active view of the given Revit **doc**.

```
public static MbResult<DirectShape> VisualizeCurveLoop2D(this CurveLoop curveLoop, Document doc, Color color = null)
```

Parameters

**curveLoop** CurveLoop

The CurveLoop to visualize.

**doc** Document

The current Revit document containing the active view.

**color** Color

Optional outline color; defaults to red if not provided.

## Returns

[MbResult](#)<DirectShape>

An [MbResult<T>](#):

- **Success:** Contains the created DirectShape.
- **Failure:** "CurveLoop is null."
- **Failure:** "Document or active view is null."
- **Failure:** "Failed to visualize CurveLoop: [exception message]"

## VisualizeCurveLoops2D(IEnumerable<CurveLoop>, Document, Color)

Visualizes a list of CurveLoop objects as 2D outlines using DirectShape elements in the active view of the given Revit `doc`.

```
public static MbResult<List<DirectShape>> VisualizeCurveLoops2D(this IEnumerable<CurveLoop>
curveLoops, Document doc, Color color = null)
```

## Parameters

`curveLoops` [IEnumerable](#)<CurveLoop>

The collection of CurveLoop instances to visualize.

`doc` Document

The current Revit document containing the active view.

`color` Color

Optional outline color; defaults to red if not provided.

## Returns

[MbResult](#)<[List](#)<DirectShape>>

An [MbResult<T>](#) where `T` is [List<T>](#):

- **Success:** Contains a list of created DirectShape elements.
- **Failure:** "CurveLoop collection is null."
- **Failure:** "Document or active view is null."
- **Failure:** "Failed to visualize multiple CurveLoops: [exception message]"



# Class CurveVisualizerExtensions

Namespace: [MBF.Revit.Geometry.Extensions](#)

Assembly: MBF.Revit.Geometry.dll

Provides extension methods for visualizing Autodesk.Revit.DB.Curve instances as 2D geometry in Revit, either using DirectShape elements in model views or Autodesk.Revit.DB.DetailCurve elements in drafting views.

```
public static class CurveVisualizerExtensions
```

## Inheritance

[object](#) ← CurveVisualizerExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### CreateCurveWithDetailCurve(Curve, Document, ViewDrafting, Color)

Creates a Autodesk.Revit.DB.DetailCurve element in a Autodesk.Revit.DB.ViewDrafting to represent the given Autodesk.Revit.DB.Curve.

```
public static MbResult<DetailCurve> CreateCurveWithDetailCurve(this Curve curve, Document doc, ViewDrafting draftingView, Color color = null)
```

## Parameters

**curve** Curve

The Autodesk.Revit.DB.Curve to draw.

**doc** Document

The current Revit document.

## `draftingView` ViewDrafting

The drafting view where the detail curve will be placed.

### `color` Color

Optional line color; defaults to red if not provided.

Returns

### [MbResult<DetailCurve>](#)

An [MbResult<T>](#):

- **Success:** Contains the created Autodesk.Revit.DB.DetailCurve element.
- **Failure:** "Curve is null."
- **Failure:** "Document or drafting view is null."
- **Failure:** "Failed to create detail curve: [exception message]"

## VisualizeCurve2D(Curve, Document, Color)

Visualizes a single Autodesk.Revit.DB.Curve as 2D geometry using a DirectShape element in the active view of the given Revit `doc`.

```
public static MbResult<DirectShape> VisualizeCurve2D(this Curve curve, Document doc, Color  
color = null)
```

Parameters

### `curve` Curve

The Autodesk.Revit.DB.Curve to visualize.

### `doc` Document

The current Revit document containing the active view.

### `color` Color

Optional outline color; defaults to red if not provided.

Returns

### [MbResult<DirectShape>](#)

An [MbResult<T>](#):

- **Success:** Contains the created DirectShape element.
- **Failure:** "Curve is null."
- **Failure:** "Document or active view is null."
- **Failure:** "Failed to visualize curve: [exception message]"

## VisualizeCurves2D(IEnumerable<Curve>, Document, Color)

Visualizes a list of Autodesk.Revit.DB.Curve instances as 2D outlines using DirectShape elements in the active view of the given Revit [doc](#).

```
public static MbResult<List<DirectShape>> VisualizeCurves2D(this IEnumerable<Curve> curves,  
Document doc, Color color = null)
```

### Parameters

[curves](#) [IEnumerable](#)<Curve>

The collection of Autodesk.Revit.DB.Curve instances to visualize.

[doc](#) Document

The current Revit document containing the active view.

[color](#) Color

Optional outline color; defaults to red if not provided.

### Returns

[MbResult](#)<[List](#)<DirectShape>>

An [MbResult<T>](#) where T is [List<T>](#):

- **Success:** Contains a list of created DirectShape elements.
- **Failure:** "Curve collection is null."
- **Failure:** "Document or active view is null."
- **Failure:** "Failed to visualize curves: [exception message]"

# Class DirectShapeVisualization

Namespace: [MBF.Revit.Geometry.Extensions](#)

Assembly: MBF.Revit.Geometry.dll

Provides methods for creating and managing DirectShape elements in a Revit document.

```
public static class DirectShapeVisualization
```

## Inheritance

[object](#) ← DirectShapeVisualization

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes utility methods for generating geometric shapes, such as spheres, and placing them in the Revit model as DirectShape elements. These methods facilitate the creation of reusable shape types and instances, leveraging Revit's geometry creation utilities.

## Methods

### CreateSphere(Document, XYZ, double)

Creates a spherical DirectShape element in the Revit document using revolved geometry.

```
public static DirectShape CreateSphere(Document doc, XYZ center, double radius)
```

#### Parameters

**doc** Document

The active Revit document where the shape will be created.

**center** XYZ

The center point of the sphere in model coordinates.

`radius` `double`

The radius of the sphere in Revit units (feet).

Returns

`DirectShape`

A `DirectShape` element representing the sphere, or `null` if the geometry could not be defined due to an invalid frame.

# Class PathDVisualizerExtensions

Namespace: [MBF.Revit.Geometry.Extensions](#)

Assembly: MBF.Revit.Geometry.dll

Provides extension methods for visualizing Clipper2Lib.PathD and Clipper2Lib.PathsD objects as 2D outlines in a Revit document using DirectShape elements.

```
public static class PathDVisualizerExtensions
```

## Inheritance

[object](#) ← PathDVisualizerExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

These methods allow developers to create visual representations of 2D paths in the active view of a Revit document. The visualizations are rendered as outlines on the XY plane and can be customized with an optional color parameter.

## Methods

### VisualizePathD2D(PathD, Document, Color)

Visualizes a single Clipper2Lib.PathD as a 2D outline using a DirectShape in the active view of the specified Revit document.

```
public static void VisualizePathD2D(this PathD path, Document doc, Color color = null)
```

#### Parameters

**path** PathD

The Clipper2Lib.PathD to visualize in 2D (XY plane).

**doc** Document

The Revit Autodesk.Revit.DB.Document where the path will be visualized.

#### **color** Color

Optional color for the path outline. Defaults to red if not specified.

## VisualizePathsD2D(PathsD, Document, Color)

Visualizes each Clipper2Lib.PathD in a Clipper2Lib.PathsD collection as a 2D outline using DirectShape elements.

```
public static void VisualizePathsD2D(this PathsD paths, Document doc, Color color = null)
```

### Parameters

#### **paths** PathsD

The Clipper2Lib.PathsD collection to visualize in 2D (XY plane).

#### **doc** Document

The Revit Autodesk.Revit.DB.Document where the paths will be visualized.

#### **color** Color

Optional color for the path outlines. Defaults to red if not specified.

# Class XyzVisualizationsExtensions

Namespace: [MBF.Revit.Geometry.Extensions](#)

Assembly: MBF.Revit.Geometry.dll

Provides extension methods for visualizing Autodesk.Revit.DB.XYZ points as 3D objects in a Revit document.

```
public static class XyzVisualizationsExtensions
```

## Inheritance

[object](#) ← XyzVisualizationsExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

These methods allow users to create visual representations of Autodesk.Revit.DB.XYZ points in the active view of a Revit document. The visualizations are implemented using DirectShape elements, with optional color customization.

## Methods

### VisualizePoint(XYZ, Document, Color)

Visualizes an Autodesk.Revit.DB.XYZ point as a 3D point object using a DirectShape in the active view.

```
public static void VisualizePoint(this XYZ point, Document doc, Color color = null)
```

#### Parameters

**point** XYZ

The point to visualize.

**doc** Document

The Revit document where the shape will be created.

**color** Color

Optional color override. Defaults to red if null.

## VisualizePointAsElement(XYZ, Document, Color)

Visualizes an Autodesk.Revit.DB.XYZ point as a 3D geometry using a DirectShape element in the active view, and returns the created element wrapped in [MbResult<T>](#).

```
public static MbResult<Element> VisualizePointAsElement(this XYZ point, Document doc, Color  
color = null)
```

### Parameters

**point** XYZ

The point to visualize.

**doc** Document

The Revit document where the shape will be created.

**color** Color

Optional color override. Defaults to red if not specified.

### Returns

[MbResult<Element>](#)

A [MbResult<T>](#):

- **Success:** The created DirectShape element.
- **Failure:** "Point, document, or active view is null."
- **Failure:** "Exception: [message]"

# Namespace MBF.Revit.Geometry.Utils

## Classes

### [MbXyzUtils](#)

Provides utility methods for working with 2D geometry, including operations on polygons and line segments.

### [PolygonUtils](#)

Provides utility methods for working with 2D polygons in the XY plane.

# Class MbXyzUtils

Namespace: [MBF.Revit.Geometry.Utils](#)

Assembly: MBF.Revit.Geometry.dll

Provides utility methods for working with 2D geometry, including operations on polygons and line segments.

```
public static class MbXyzUtils
```

## Inheritance

[object](#) ← MbXyzUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes methods for detecting self-intersections in polygons, determining line segment intersections, and sorting points in a clockwise order. It is designed to work with 2D geometry represented by Autodesk.Revit.DB.XYZ points, where the Z-coordinate is ignored in calculations.

## Methods

### DoLinesIntersect2D(Line, Line)

Determines whether two Revit Autodesk.Revit.DB.Line objects intersect in 2D (ignoring Z).

```
public static bool DoLinesIntersect2D(Line line1, Line line2)
```

#### Parameters

**line1** Line

First line.

**line2** Line

Second line.

Returns

[bool](#)

[true](#) if the lines intersect in 2D; otherwise, [false](#).

## DoLinesIntersect2D(XYZ, XYZ, XYZ, XYZ)

Determines whether two line segments in 2D space intersect.

```
public static bool DoLinesIntersect2D(XYZ segment1Start, XYZ segment1End, XYZ segment2Start,  
XYZ segment2End)
```

Parameters

**segment1Start** XYZ

The starting point of the first line segment.

**segment1End** XYZ

The ending point of the first line segment.

**segment2Start** XYZ

The starting point of the second line segment.

**segment2End** XYZ

The ending point of the second line segment.

Returns

[bool](#)

[true](#) if the two line segments intersect; otherwise, [false](#).

Remarks

This method uses a counter-clockwise orientation test to determine if the line segments intersect. The segments are considered to intersect if they share any point in common, including endpoints.

## HasSelfIntersections(List<XYZ>)

Determines whether a list of Autodesk.Revit.DB.XYZ points forming a polygon has any self-intersections.

```
public static bool HasSelfIntersections(List<XYZ> points)
```

### Parameters

**points** [List](#)<XYZ>

A list of points representing the polygon vertices in order.

### Returns

[bool](#)

**true** if any two non-adjacent edges of the polygon intersect; otherwise, **false**.

### Remarks

The method performs three checks:

1. Checks all non-adjacent segment pairs within the polygon to detect internal intersections.
2. Checks if the most recently added segment intersects any existing (non-adjacent) segment.
3. Checks if the closing segment (from the last point to the first) intersects any interior segment (excluding adjacent ones).

## SortPointsClockwise(List<XYZ>)

Sorts a list of Autodesk.Revit.DB.XYZ points in clockwise order around their centroid (in 2D plane).

```
public static List<XYZ> SortPointsClockwise(List<XYZ> points)
```

### Parameters

**points** [List](#)<XYZ>

The list of points to sort.

### Returns

[List](#)<XYZ>

A new list of points sorted clockwise.

# Class PolygonUtils

Namespace: [MBF.Revit.Geometry.Utils](#)

Assembly: MBF.Revit.Geometry.dll

Provides utility methods for working with 2D polygons in the XY plane.

```
public static class PolygonUtils
```

## Inheritance

[object](#) ← PolygonUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class includes methods for determining spatial relationships between points and polygons, such as checking whether a point lies inside a polygon. All calculations are performed in the XY plane, and Z-values of input points are ignored.

## Methods

### IsPointInsidePolygon(List<XYZ>, XYZ)

Determines whether a given point lies inside a 2D polygon using the ray-casting algorithm (in XY plane).

```
public static MbResult<bool> IsPointInsidePolygon(List<XYZ> polygon, XYZ point)
```

#### Parameters

**polygon** [List](#)<XYZ>

The list of polygon vertices (at least 3 points required). Z-values are ignored.

**point** XYZ

The point to check. Z-value is ignored.

## Returns

[MbResult<bool>](#)

An [MbResult<T>](#) containing:

- **true** if the point lies inside the polygon.
- **false** if the point lies outside or on the edge.
- An error message if the input is invalid or an exception occurs.

# Namespace MBF.Revit.Geometry.Voronoi.Models

## Classes

### [Edge](#)

Represents an edge in the Voronoi diagram, defined by a line equation and associated sites.

### [GraphEdge](#)

Represents a graphical edge in the Voronoi diagram, defined by two endpoints and the indices of the two sites it separates.

### [Halfedge](#)

Represents a half-edge in the Voronoi diagram, used primarily in Fortune's algorithm for managing the beach line and event queue.

### [Point](#)

Represents a point in 2D space.

### [Site](#)

Represents a site (or generator point) used in the Voronoi diagram, typically associated with a cell in the diagram.

### [SiteSorterYX](#)

Provides a comparison mechanism for sorting [Site](#) instances. Sorts first by Y-coordinate, then by X-coordinate if Y values are equal. Useful for processing sites in Fortune's algorithm.

### [Voronoi](#)

Represents a Voronoi diagram generator.

# Class Edge

Namespace: [MBF.Revit.Geometry.Voronoi.Models](#)

Assembly: MBF.Revit.Geometry.dll

Represents an edge in the Voronoi diagram, defined by a line equation and associated sites.

```
public class Edge
```

## Inheritance

[object](#) ← Edge

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Constructors

## Edge()

Initializes a new instance of the [Edge](#) class.

```
public Edge()
```

# Fields

## A

Coefficient 'a' in the line equation of the edge ( $a * x + b * y = c$ ).

```
public double A
```

## Field Value

[double](#)

## B

Coefficient 'b' in the line equation of the edge ( $a * x + b * y = c$ ).

```
public double B
```

Field Value

[double](#)

## C

Constant term 'c' in the line equation of the edge ( $a * x + b * y = c$ ).

```
public double C
```

Field Value

[double](#)

## Edgenbr

Unique identifier for the edge.

```
public int Edgenbr
```

Field Value

[int](#)

## Ep

Endpoints of the edge (can be null if the edge is a ray or still under construction).

```
public Site[] Ep
```

Field Value

[Site\[\]](#)

## Reg

Sites that the edge bisects; these are the two sites whose Voronoi cells are separated by this edge.

```
public Site[] Reg
```

Field Value

[Site\[\]](#)

# Class GraphEdge

Namespace: [MBF.Revit.Geometry.Voronoi.Models](#)

Assembly: MBF.Revit.Geometry.dll

Represents a graphical edge in the Voronoi diagram, defined by two endpoints and the indices of the two sites it separates.

```
public class GraphEdge
```

## Inheritance

[object](#) ← GraphEdge

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Constructors

## GraphEdge()

```
public GraphEdge()
```

# Fields

## site1

The index of the first site that this edge separates.

```
public int site1
```

## Field Value

[int](#)

## site2

The index of the second site that this edge separates.

```
public int site2
```

Field Value

[int](#)

## x1

The X-coordinate of the starting point of the edge.

```
public double x1
```

Field Value

[double](#)

## x2

The X-coordinate of the ending point of the edge.

```
public double x2
```

Field Value

[double](#)

## y1

The Y-coordinate of the starting point of the edge.

```
public double y1
```

Field Value

[double](#)

## y2

The Y-coordinate of the ending point of the edge.

```
public double y2
```

Field Value

[double](#)

# Class Halfedge

Namespace: [MBF.Revit.Geometry.Voronoi.Models](#)

Assembly: MBF.Revit.Geometry.dll

Represents a half-edge in the Voronoi diagram, used primarily in Fortune's algorithm for managing the beach line and event queue.

```
public class Halfedge
```

## Inheritance

[object](#) ← Halfedge

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Constructors

## Halfedge()

Initializes a new instance of the [Halfedge](#) class.

```
public Halfedge()
```

# Fields

## ELedge

The edge associated with this half-edge.

```
public Edge ELedge
```

## Field Value

[Edge](#)

## ELleft

The left neighboring half-edge in the edge list.

```
public Halfedge ELleft
```

Field Value

[Halfedge](#)

## ELpm

Side indicator: 0 for left, 1 for right. Determines which region the half-edge belongs to.

```
public int ELpm
```

Field Value

[int↗](#)

## ELright

The right neighboring half-edge in the edge list.

```
public Halfedge ELright
```

Field Value

[Halfedge](#)

## PQnext

The next half-edge in the priority queue.

```
public Halfedge PQnext
```

Field Value

[Halfedge](#)

## deleted

Indicates whether this half-edge has been marked for deletion.

```
public bool deleted
```

Field Value

[bool](#) ↗

## vertex

The vertex where this half-edge ends (used in the priority queue for upcoming events).

```
public Site vertex
```

Field Value

[Site](#)

## ystar

The Y-coordinate of the event associated with this half-edge in the priority queue.

```
public double ystar
```

Field Value

[double](#) ↗

# Class Point

Namespace: [MBF.Revit.Geometry.Voronoi.Models](#)

Assembly: MBF.Revit.Geometry.dll

Represents a point in 2D space.

```
public class Point
```

## Inheritance

[object](#) ← Point

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Constructors

## Point()

Initializes a new instance of the [Point](#) class.

```
public Point()
```

# Fields

## X

The x-coordinate of the point.

```
public double x
```

## Field Value

[double](#)

y

The y-coordinate of the point.

```
public double y
```

Field Value

[double](#)

## Methods

**SetPoint(double, double)**

Sets the coordinates of the point.

```
public void SetPoint(double x, double y)
```

Parameters

x [double](#)

The x-coordinate.

y [double](#)

The y-coordinate.

# Class Site

Namespace: [MBF.Revit.Geometry.Voronoi.Models](#)

Assembly: MBF.Revit.Geometry.dll

Represents a site (or generator point) used in the Voronoi diagram, typically associated with a cell in the diagram.

```
public class Site
```

## Inheritance

[object](#) ← Site

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Constructors

## Site()

Initializes a new instance of the [Site](#) class.

```
public Site()
```

# Fields

## coord

The coordinates of the site in 2D space.

```
public Point coord
```

## Field Value

[Point](#)

## sitenbr

A unique identifier for the site.

```
public int sitenbr
```

Field Value

[int ↗](#)

# Class SiteSorterYX

Namespace: [MBF.Revit.Geometry.Voronoi.Models](#)

Assembly: MBF.Revit.Geometry.dll

Provides a comparison mechanism for sorting [Site](#) instances. Sorts first by Y-coordinate, then by X-coordinate if Y values are equal. Useful for processing sites in Fortune's algorithm.

```
public class SiteSorterYX : IComparer<Site>
```

## Inheritance

[object](#) ← SiteSorterYX

## Implements

[IComparer](#) <[Site](#)>

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

SiteSorterYX()

```
public SiteSorterYX()
```

## Methods

Compare(Site, Site)

Compares two [Site](#) objects based on their Y and X coordinates.

```
public int Compare(Site p1, Site p2)
```

## Parameters

**p1** [Site](#)

The first site to compare.

**p2** [Site](#)

The second site to compare.

Returns

[int↗](#)

-1 if **p1** comes before **p2**, 1 if it comes after, 0 if they are equal.

# Class Voronoi

Namespace: [MBF.Revit.Geometry.Voronoi.Models](#)

Assembly: MBF.Revit.Geometry.dll

Represents a Voronoi diagram generator.

```
public class Voronoi
```

## Inheritance

[object](#) ← Voronoi

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Constructors

## Voronoi(double)

Initializes a new instance of the [Voronoi](#) class.

```
public Voronoi(double minDistanceBetweenSites)
```

## Parameters

`minDistanceBetweenSites` [double](#)

The minimum distance between sites.

# Methods

## generateVoronoi(double[], double[], double, double, double, double)

Generates the Voronoi diagram.

```
public List<GraphEdge> generateVoronoi(double[] xValuesIn, double[] yValuesIn, double minX,  
double maxX, double minY, double maxY)
```

## Parameters

**xValuesIn** [double](#)[]

The x-coordinates of the input sites.

**yValuesIn** [double](#)[]

The y-coordinates of the input sites.

**minX** [double](#)

The minimum x-coordinate of the bounding box.

**maxX** [double](#)

The maximum x-coordinate of the bounding box.

**minY** [double](#)

The minimum y-coordinate of the bounding box.

**maxY** [double](#)

The maximum y-coordinate of the bounding box.

## Returns

[List](#)<[GraphEdge](#)>

A list of [GraphEdge](#) representing the edges of the Voronoi diagram.

# Namespace MBF.Revit.Geometry.Voronoi.Utils

## Classes

### [MbVoronoiAlgorithmHelpers](#)

Provides helper methods for generating Voronoi diagrams within a specified polygonal boundary.

# Class MbVoronoiAlgorithmHelpers

Namespace: [MBF.Revit.Geometry.Voronoi.Utils](#)

Assembly: MBF.Revit.Geometry.dll

Provides helper methods for generating Voronoi diagrams within a specified polygonal boundary.

```
[LogAspect]  
public static class MbVoronoiAlgorithmHelpers
```

## Inheritance

[object](#) ← MbVoronoiAlgorithmHelpers

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class contains methods to create Voronoi diagrams based on a set of input points and a polygonal boundary. The generated Voronoi diagram is clipped to the boundary of the polygon, and the results are drawn in the active Revit view.

## Methods

### CreateVoronoiDiagramAsync(List<XYZ>, List<XYZ>)

Generates a Voronoi diagram based on the provided input points and clips it to the specified polygon boundary.

```
public static Task CreateVoronoiDiagramAsync(List<XYZ> inputPoints, List<XYZ> polygonPoints)
```

#### Parameters

**inputPoints** [List](#)<XYZ>

A list of points representing the input data for generating the Voronoi diagram. Each point must have valid X and Y coordinates.

## **polygonPoints** [List](#)<XYZ>

A list of points defining the vertices of the polygon boundary. The polygon must be closed, with the first and last points being the same.

Returns

## [Task](#)

Remarks

This method creates a Voronoi diagram by dividing the space into regions around each input point, and then clips the resulting diagram to the specified polygon boundary. The diagram is drawn as detail curves in the active view of the Revit document.

The method performs the following steps:

- Converts the polygon boundary into a geometric representation.
- Generates Voronoi edges based on the input points.
- Clips the edges to the polygon boundary and draws them in the active Revit view.
- Calculates the coverage area for each input point within the polygon boundary.

# Namespace MBFWpfToolkit

## Classes

### [MbWindow](#)

Represents a customizable WPF window with extended properties and features, including theming, language selection, and plugin-related configurations.

# Class MbWindow

Namespace: [MBFWpfToolkit](#)

Assembly: MBFWpfToolkit.dll

Represents a customizable WPF window with extended properties and features, including theming, language selection, and plugin-related configurations.

```
public class MbWindow : Window, INotifyPropertyChanged
```

Inheritance

[object](#) ← MbWindow

Implements

[INotifyPropertyChanged](#)

Derived

[HelpWizard](#), [MbMessageBox](#)

## Constructors

### MbWindow()

Initializes a new instance of the [MbWindow](#) class.

```
public MbWindow()
```

## Remarks

This constructor sets up event handlers for the Loaded and Closed events. It ensures that main resources are cached and reused across windows.

## Fields

### IsThemeToggleVisibleProperty

Represents the dependency property that identifies whether the theme toggle is visible.

```
public static readonly DependencyProperty IsThemeToggleVisibleProperty
```

Field Value

[DependencyProperty](#)

## PluginDescriptionProperty

Identifies the PluginDescription dependency property.

```
public static readonly DependencyProperty PluginDescriptionProperty
```

Field Value

[DependencyProperty](#)

## PluginNameProperty

Identifies the PluginName dependency property.

```
public static readonly DependencyProperty PluginNameProperty
```

Field Value

[DependencyProperty](#)

## PluginVersionProperty

Identifies the PluginVersion dependency property.

```
public static readonly DependencyProperty PluginVersionProperty
```

Field Value

[DependencyProperty](#)

## PrimaryBrushProperty

Identifies the PrimaryBrush dependency property.

```
public static readonly DependencyProperty PrimaryBrushProperty
```

Field Value

[DependencyProperty](#)

## ShowLanguagesComboBoxProperty

Identifies the ShowLanguagesComboBox dependency property.

```
public static readonly DependencyProperty ShowLanguagesComboBoxProperty
```

Field Value

[DependencyProperty](#)

## ShowLogoProperty

Identifies the [ShowLogo](#) dependency property.

```
public static readonly DependencyProperty ShowLogoProperty
```

Field Value

[DependencyProperty](#)

## ShowPluginNameProperty

Identifies the [ShowPluginName](#) dependency property.

```
public static readonly DependencyProperty ShowPluginNameProperty
```

Field Value

[DependencyProperty](#)

# ShowPluginVersionProperty

Identifies the [ShowPluginVersion](#) dependency property.

```
public static readonly DependencyProperty ShowPluginVersionProperty
```

Field Value

[DependencyProperty](#)

## Properties

### IsDark

Gets or sets a value indicating whether the window should use a dark theme or not.

```
public bool IsDark { get; set; }
```

Property Value

[bool](#)

### IsThemeToggleVisible

Gets or sets a value indicating whether the theme toggle control is visible.

```
public bool IsThemeToggleVisible { get; set; }
```

Property Value

[bool](#)

### Remarks

By default, the theme toggle control is visible unless explicitly set to false.

## this[string]

Represents the current instance of the class. Use 'this' to refer explicitly to the calling object in methods, constructors, or property accessors. It is useful to resolve naming conflicts between instance members and method or constructor arguments.

```
public string this[string key] { get; }
```

## Parameters

**key** [string](#)

## Property Value

[string](#)

## MainResources

Gets or sets the shared resource dictionary that can be used to cache and reuse resources across multiple instances of [MbWindow](#).

```
public static ResourceDictionary? MainResources { get; set; }
```

## Property Value

[ResourceDictionary](#)

## PluginDescription

Identifies the [PluginDescription](#) dependency property.

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

## Property Value

[string](#)

## PluginName

Identifies the [PluginName](#) dependency property.

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

Property Value

[string](#) ↗

## PluginVersion

Identifies the [PluginVersion](#) dependency property.

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

Property Value

[string](#) ↗

## PrimaryBrush

Identifies the [PrimaryBrush](#) dependency property.

```
public Color PrimaryBrush { get; set; }
```

Property Value

[Color](#) ↗

## SelectedLanguage

Gets or sets the selected language for the window.

```
public CultureInfo SelectedLanguage { get; set; }
```

Property Value

[CultureInfo](#) ↗

## ShowLanguagesComboBox

Identifies the [ShowLanguagesComboBox](#) dependency property.

```
public bool ShowLanguagesComboBox { get; set; }
```

Property Value

[bool](#) ↗

## ShowLogo

Identifies the [ShowLogo](#) dependency property.

```
public bool ShowLogo { get; set; }
```

Property Value

[bool](#) ↗

## ShowPluginName

Identifies the [ShowPluginName](#) dependency property.

```
public bool ShowPluginName { get; set; }
```

Property Value

[bool](#) ↗

## ShowPluginVersion

Identifies the [ShowPluginVersion](#) dependency property.

```
public bool ShowPluginVersion { get; set; }
```

Property Value

[bool](#) ↗

# SupportedLanguages

Gets the collection of supported languages available for the application.

```
public ObservableCollection<CultureInfo> SupportedLanguages { get; }
```

## Property Value

[ObservableCollection](#)<[CultureInfo](#)>

# Methods

## ApplyThemeForElement(FrameworkElement, string)

Applies a theme to the specified framework element based on the provided theme name.

```
public static void ApplyThemeForElement(FrameworkElement element, string darkOrLight)
```

### Parameters

**element** [FrameworkElement](#)

The framework element to which the theme will be applied.

**darkOrLight** [string](#)

The name of the theme, typically indicating dark or light mode.

## GetAllChildrenOfType<T>(DependencyObject)

Retrieves all children of a specified type from a given dependency object in the visual tree.

```
public static IEnumerable<T> GetAllChildrenOfType<T>(DependencyObject parent) where T : DependencyObject
```

### Parameters

**parent** [DependencyObject](#)

The parent dependency object to search within.

Returns

[IEnumerable](#)<T>

Type Parameters

T

The type of children to find.

## GetPrimaryBrush()

Gets the primary brush color.

```
protected static Color GetPrimaryBrush()
```

Returns

[Color](#)

The primary brush color.

## InitializeWindow()

Performs static initialization for the [MbWindow](#). Applies the primary brush color, sets window properties, configures non-client UI content, and ensures shared resources are loaded into [MainResources](#).

```
protected void InitializeWindow()
```

## OnIsDarkChanged(bool, bool)

Called when IsDark property changes

```
protected virtual void OnIsDarkChanged(bool oldValue, bool newValue)
```

Parameters

**oldValue** [bool ↗](#)

Previous value

**newValue** [bool ↗](#)

New value

## OnPropertyChanged(string?)

Invoked when a property value changes.

```
protected virtual void OnPropertyChanged(string? propertyName = null)
```

### Parameters

**propertyName** [string ↗](#)

The name of the property that changed. Automatically set by the caller if not supplied.

## SetField<T>(ref T, T, string?)

Updates the specified field with a new value and raises the PropertyChanged event if the value changes.

```
protected bool SetField<T>(ref T field, T value, string? propertyName = null)
```

### Parameters

**field** T

A reference to the field being updated.

**value** T

The new value to assign to the field.

**propertyName** [string ↗](#)

The name of the property. This parameter is optional and is automatically supplied by the CallerMemberName attribute.

Returns

[bool](#)

True if the field value was updated; otherwise, false.

Type Parameters

T

The type of the field and value.

## ShowSprite()

Displays the [MicroBimSprite](#) overlay window, which can act as a floating mascot or animation on the screen. Initializes an instance of the [MicroBimSprite](#), stores it internally, and invokes its [Show\(\)](#) method to render it.

```
public void ShowSprite()
```

## Events

### IsDarkChanged

Event raised when IsDark property changes

```
public event EventHandler<bool> IsDarkChanged
```

Event Type

[EventHandler](#) <[bool](#)>

### PropertyChanged

Occurs when a property value changes.

```
public event PropertyChangedEventHandler PropertyChanged
```

Event Type

[PropertyChangedEventHandler](#)

# Namespace MBFWpfToolkit.AttachedProperties

## Classes

### [ComboBoxSearchBehavior](#)

Provides an attached behavior to enable search functionality in a [ComboBox](#), allowing live filtering of items based on user input.

### [HelpAssistant](#)

Provides attached properties to define step-based help or onboarding instructions for UI elements. Elements can be dynamically registered and retrieved in a specific order based on their step number.

### [LocalizationHelper](#)

Provides localization support for WPF controls using attached properties. When [LangKeyProperty](#) is set on a control, the corresponding localized value will be applied automatically.

### [MicroBimThemeBehavior](#)

### [ThemeBehavior](#)

Provides attached properties and behaviors for managing themes in a WPF application.

### [ThemeUtils](#)

Provides utility methods for handling theme-related operations on WPF [FrameworkElement](#) instances.

# Class ComboBoxSearchBehavior

Namespace: [MBFWpfToolkit.AttachedProperties](#)

Assembly: MBFWpfToolkit.dll

Provides an attached behavior to enable search functionality in a [ComboBox](#), allowing live filtering of items based on user input.

```
public static class ComboBoxSearchBehavior
```

## Inheritance

[object](#) ← ComboBoxSearchBehavior

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Fields

## IsSearchEnabledProperty

Identifies the IsSearchEnabled dependency property. This property can be attached to a [ComboBox](#) to enable or disable live search filtering functionality.

```
public static readonly DependencyProperty IsSearchEnabledProperty
```

## Field Value

[DependencyProperty](#)

true if the search functionality is enabled for the associated ComboBox; otherwise, false. The default value is false.

## Remarks

When this property is set to true, the ComboBox supports a live search filtering mechanism for its items based on the current input text. It also modifies the behavior of the ComboBox, making it editable, disabling built-in text search, and keeping the dropdown open during edits. When set to false, the standard ComboBox behavior is restored.

# Methods

## GetFilteredItems(DependencyObject)

Gets the filtered items associated with the specified [DependencyObject](#). The filtered items are updated dynamically as the user types into the search-enabled ComboBox.

```
public static ObservableCollection<string> GetFilteredItems(DependencyObject obj)
```

### Parameters

**obj** [DependencyObject](#)

The [DependencyObject](#) for which to retrieve the filtered items.

### Returns

[ObservableCollection](#)<[string](#)>

An [ObservableCollection](#)<[T](#)> of strings representing the currently filtered items.

## GetIsSearchEnabled(DependencyObject)

Gets the value of the IsSearchEnabled attached property for a specified [DependencyObject](#).

```
public static bool GetIsSearchEnabled(DependencyObject obj)
```

### Parameters

**obj** [DependencyObject](#)

The [DependencyObject](#) from which to read the attached property value.

### Returns

[bool](#)

A boolean value indicating whether the live search behavior is enabled on the specified [DependencyObject](#).

## SetFilteredItems(DependencyObject, ObservableCollection<string>)

Sets the filtered collection of items to be displayed in the [ComboBox](#) based on the current search behavior.

```
public static void SetFilteredItems(DependencyObject obj, ObservableCollection<string>  
value)
```

### Parameters

**obj** [DependencyObject](#)

The [DependencyObject](#) to set the filtered items for.

**value**  [ObservableCollection<string>](#)

The  [ObservableCollection<T>](#) of filtered items to set.

## SetIsSearchEnabled(DependencyObject, bool)

Sets the value of the  [IsSearchEnabledProperty](#) attached property, enabling or disabling live search filtering on a [ComboBox](#).

```
public static void SetIsSearchEnabled(DependencyObject obj, bool value)
```

### Parameters

**obj** [DependencyObject](#)

The dependency object to which the property is attached.

**value** [bool](#)

A boolean value indicating whether the search feature should be enabled or disabled.

# Class HelpAssistant

Namespace: [MBFWpfToolkit.AttachedProperties](#)

Assembly: MBFWpfToolkit.dll

Provides attached properties to define step-based help or onboarding instructions for UI elements. Elements can be dynamically registered and retrieved in a specific order based on their step number.

```
public class HelpAssistant
```

## Inheritance

[object](#) ← HelpAssistant

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### HelpAssistant()

```
public HelpAssistant()
```

## Fields

### AutoRegisterProperty

Identifies the AutoRegister attached property. This property is used only to trigger element registration dynamically.

```
public static readonly DependencyProperty AutoRegisterProperty
```

## Field Value

[DependencyProperty](#)

## StepNumberProperty

Identifies the StepNumber attached property. Represents the sequence order of the help step.

```
public static readonly DependencyProperty StepNumberProperty
```

Field Value

[DependencyProperty](#)

## StepTextProperty

Identifies the StepText attached property. Describes the instructional text for the help step.

```
public static readonly DependencyProperty StepTextProperty
```

Field Value

[DependencyProperty](#)

## Methods

### GetAutoRegister(DependencyObject)

Gets the AutoRegister value of the element.

```
public static bool GetAutoRegister(DependencyObject obj)
```

Parameters

**obj** [DependencyObject](#)

Returns

[bool](#)

### GetOrderedSteps()

Retrieves all registered UI elements with valid step numbers, sorted in ascending order.

```
public static List<FrameworkElement> GetOrderedSteps()
```

Returns

[List](#)<[FrameworkElement](#)>

A sorted list of registered [FrameworkElement](#)s for the help assistant.

## GetStepNumber(DependencyObject)

Gets the step number of the element.

```
public static int GetStepNumber(DependencyObject obj)
```

Parameters

**obj** [DependencyObject](#)

Returns

[int](#)

## GetStepText(DependencyObject)

Gets the step text of the element.

```
public static string GetStepText(DependencyObject obj)
```

Parameters

**obj** [DependencyObject](#)

Returns

[string](#)

## SetAutoRegister(DependencyObject, bool)

Sets the AutoRegister value of the element.

```
public static void SetAutoRegister(DependencyObject obj, bool value)
```

Parameters

**obj** [DependencyObject](#)

**value** [bool](#)

## SetStepNumber(DependencyObject, int)

Sets the step number of the element.

```
public static void SetStepNumber(DependencyObject obj, int value)
```

Parameters

**obj** [DependencyObject](#)

**value** [int](#)

## SetStepText(DependencyObject, string)

Sets the step text of the element.

```
public static void SetStepText(DependencyObject obj, string value)
```

Parameters

**obj** [DependencyObject](#)

**value** [string](#)

# Class LocalizationHelper

Namespace: [MBFWpfToolkit.AttachedProperties](#)

Assembly: MBFWpfToolkit.dll

Provides localization support for WPF controls using attached properties. When [LangKeyProperty](#) is set on a control, the corresponding localized value will be applied automatically.

```
public static class LocalizationHelper
```

## Inheritance

[object](#) ← LocalizationHelper

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Fields

## LangKeyProperty

Identifies the [LangKey](#) attached property. This property is used to bind a localization key to a control.

```
public static readonly DependencyProperty LangKeyProperty
```

## Field Value

[DependencyProperty](#)

# Methods

## GetLangKey(DependencyObject)

Gets the value of the [LangKey](#) attached property.

```
public static string GetLangKey(DependencyObject obj)
```

Parameters

**obj** [DependencyObject](#)

Returns

[string](#)

## SetLangKey(DependencyObject, string)

Sets the value of the [LangKey](#) attached property.

```
public static void SetLangKey(DependencyObject obj, string value)
```

Parameters

**obj** [DependencyObject](#)

**value** [string](#)

# Class MicroBimThemeBehavior

Namespace: [MBFWpfToolkit.AttachedProperties](#)

Assembly: MBFWpfToolkit.dll

```
public static class MicroBimThemeBehavior
```

## Inheritance

[object](#) ← MicroBimThemeBehavior

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Fields

### AnimateThemeOnLoadProperty

```
public static readonly DependencyProperty AnimateThemeOnLoadProperty
```

## Field Value

[DependencyProperty](#)

## Methods

### GetAnimateThemeOnLoad(DependencyObject)

```
public static bool GetAnimateThemeOnLoad(DependencyObject element)
```

## Parameters

element [DependencyObject](#)

Returns

bool ↗

## SetAnimateThemeOnLoad(DependencyObject, bool)

```
public static void SetAnimateThemeOnLoad(DependencyObject element, bool value)
```

Parameters

**element** DependencyObject ↗

**value** bool ↗

# Class ThemeBehavior

Namespace: [MBFWpfToolkit.AttachedProperties](#)

Assembly: MBFWpfToolkit.dll

Provides attached properties and behaviors for managing themes in a WPF application.

```
public static class ThemeBehavior
```

## Inheritance

[object](#) ← ThemeBehavior

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Fields

## ApplyThemeOnLoadProperty

Represents an attached dependency property that ensures a theme is automatically applied to a [FrameworkElement](#) when it is loaded in the WPF visual tree.

```
public static readonly DependencyProperty ApplyThemeOnLoadProperty
```

## Field Value

[DependencyProperty](#)

## Remarks

When this property is set to true on a [FrameworkElement](#), it subscribes to the element's Loaded event. Once the element is loaded, it applies the appropriate theme ('Dark' or 'Light') based on the associated [MbWindow](#) instance's theme state. The theme is applied only if the element contains resources associated with the MicroBIM theme and if the current theme differs from the desired one.

# Methods

## GetApplyThemeOnLoad(DependencyObject)

Gets the value indicating whether the theme should be applied when the associated element is loaded.

```
public static bool GetApplyThemeOnLoad(DependencyObject element)
```

### Parameters

**element** [DependencyObject](#)

The [DependencyObject](#) to retrieve the attached property value from.

### Returns

[bool](#)

True if the theme should be applied on load; otherwise, false.

## SetApplyThemeOnLoad(DependencyObject, bool)

```
public static void SetApplyThemeOnLoad(DependencyObject element, bool value)
```

### Parameters

**element** [DependencyObject](#)

The dependency object (typically a framework element) on which to set the ApplyThemeOnLoad property.

**value** [bool](#)

A boolean value indicating whether the theme will be applied when the element loads.

# Class ThemeUtils

Namespace: [MBFWpfToolkit.AttachedProperties](#)

Assembly: MBFWpfToolkit.dll

Provides utility methods for handling theme-related operations on WPF [FrameworkElement](#) instances.

```
public static class ThemeUtils
```

## Inheritance

[object](#) ← ThemeUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### ApplyThemeForElement(FrameworkElement, string)

Applies a specified theme (Dark or Light) to the given FrameworkElement.

```
public static void ApplyThemeForElement(FrameworkElement element, string darkOrLight)
```

#### Parameters

**element** [FrameworkElement](#)

The FrameworkElement to which the theme will be applied.

**darkOrLight** [string](#)

A string specifying the theme to apply. Accepted values are "Dark" or "Light".

### GetCurrentThemeForElement(FrameworkElement)

```
public static string? GetCurrentThemeForElement(FrameworkElement element)
```

## Parameters

**element** [FrameworkElement](#)

The FrameworkElement for which the current theme is to be identified. Must be non-null and have resource dictionaries.

## Returns

[string](#)

A string representing the current theme applied to the element ("Dark" or "Light"), or null if no compatible theme is found.

## HasMicroBimTheme(FrameworkElement)

```
public static bool HasMicroBimTheme(FrameworkElement element)
```

## Parameters

**element** [FrameworkElement](#)

The FrameworkElement to inspect for the MicroBIM theme.

## Returns

[bool](#)

True if the FrameworkElement's resources include the MicroBIM theme; otherwise, false.

# Namespace MBFWpfToolkit.Constants

## Enums

### [MbMessageIcon](#)

Defines icons that can be displayed in a custom message box. These icons represent different statuses or types of feedback for the user.

### [SkinType](#)

Defines available theme skins for the application UI.

# Enum MbMessageIcon

Namespace: [MBFWpfToolkit.Constants](#)

Assembly: MBFWpfToolkit.dll

Defines icons that can be displayed in a custom message box. These icons represent different statuses or types of feedback for the user.

```
public enum MbMessageIcon
```

## Fields

**Error = 16**

A white X in a circle with a red background, indicating an error or failure.

**Information = 64**

An exclamation point in a triangle with a yellow background, providing informational feedback.

**New = 512**

A trash bin or cross symbol, indicating the creation of a new item.

**None = 0**

No icon is displayed.

**Question = 48**

A question mark in a circle, prompting the user for confirmation or input.

**Remove = 1024**

A trash bin or cross symbol, indicating the removal or deletion of an item.

**Sound = 32**

A bold white X in a circle with a darker red background, indicating a critical error.

**Success = 256**

A lowercase letter 'i' in a circle with a green background, indicating success or completion.

**Warning = 128**

An exclamation point in a triangle with an orange background, warning about a potential issue.

# Enum SkinType

Namespace: [MBFWpfToolkit.Constants](#)

Assembly: MBFWpfToolkit.dll

Defines available theme skins for the application UI.

```
public enum SkinType
```

## Fields

**Dark = 1**

A dark theme with a darker background and lighter foreground elements.

**Default = 0**

The default (light or system-defined) application theme.

**Violet = 2**

A violet-colored theme for a more vibrant visual experience.

# Namespace MBFWpfToolkit.Controls

## Classes

### [MbMessageBox](#)

A custom message box window that supports MicroBIM.UI theming, icon display, and multiple button configurations.

### [MicroBimSprite](#)

Represents the MicroBIM Sprite window used for displaying custom interactive UI elements.

# Class MbMessageBox

Namespace: [MBFWpfToolkit.Controls](#)

Assembly: MBFWpfToolkit.dll

A custom message box window that supports MicroBIM.UI theming, icon display, and multiple button configurations.

```
public class MbMessageBox : MbWindow, INotifyPropertyChanged
```

## Inheritance

[object](#) ← [MbWindow](#) ← MbMessageBox

## Implements

[INotifyPropertyChanged](#)

## Inherited Members

[MbWindow.IsThemeToggleVisibleProperty](#) , [MbWindow.ShowLanguagesComboBoxProperty](#) ,  
[MbWindow.PluginNameProperty](#) , [MbWindow.PluginDescriptionProperty](#) ,  
[MbWindow.PluginVersionProperty](#) , [MbWindow.PrimaryBrushProperty](#) , [MbWindow.ShowLogoProperty](#) ,  
[MbWindow.ShowPluginNameProperty](#) , [MbWindow.ShowPluginVersionProperty](#) ,  
[MbWindow.OnIsDarkChanged\(bool, bool\)](#) , [MbWindow.GetAllChildrenOfType<T>\(DependencyObject\)](#) ,  
[MbWindow.ApplyThemeForElement\(FrameworkElement, string\)](#) , [MbWindow.InitializeWindow\(\)](#) ,  
[MbWindow.ShowSprite\(\)](#) , [MbWindow.GetPrimaryBrush\(\)](#) , [MbWindow.OnPropertyChanged\(string\)](#) ,  
[MbWindow.SetField<T>\(ref T, T, string\)](#) , [MbWindow.IsDark](#) , [MbWindow.SelectedLanguage](#) ,  
[MbWindow.SupportedLanguages](#) , [MbWindow.this\[string\]](#) , [MbWindow.IsThemeToggleVisible](#) ,  
[MbWindow.ShowLanguagesComboBox](#) , [MbWindow.PluginName](#) , [MbWindow.PluginDescription](#) ,  
[MbWindow.PluginVersion](#) , [MbWindow.PrimaryBrush](#) , [MbWindow.ShowLogo](#) ,  
[MbWindow.ShowPluginName](#) , [MbWindow.ShowPluginVersion](#) , [MbWindow.MainResources](#) ,  
[MbWindow.IsDarkChanged](#) , [MbWindow.PropertyChanged](#)

## Fields

### \_parentWindow

```
public static MbWindow _parentWindow
```

## Field Value

## Methods

### OnClosed(EventArgs)

Cleanup when window is closed

```
protected override void OnClosed(EventArgs e)
```

Parameters

e [EventArgs](#)

### Show(string, MessageBoxButton, MbMessageIcon)

Displays the message box with the specified message and button configuration.

```
public static MessageBoxResult Show(string message, MessageBoxButton button =  
MessageBoxButton.OK, MbMessageIcon icon = MbMessageIcon.None)
```

Parameters

message [string](#)

The message to display.

button [MessageBoxButton](#)

The button configuration.

icon [MbMessageIcon](#)

The icon to display.

Returns

[MessageBoxResult](#)

The result of the button clicked.

# Class MicroBimSprite

Namespace: [MBFWpfToolkit.Controls](#)

Assembly: MBFWpfToolkit.dll

Represents the MicroBIM Sprite window used for displaying custom interactive UI elements.

```
public class MicroBimSprite : Window, IComponentConnector
```

## Inheritance

[object](#) ← MicroBimSprite

## Implements

[IComponentConnector](#)

## Remarks

This window is styled using MicroBIM.UI and can be used to provide compact, reusable, or themed dialogs/popups.

## Constructors

### MicroBimSprite()

Initializes a new instance of the [MicroBimSprite](#) window. Loads the associated XAML and prepares the window for display.

```
public MicroBimSprite()
```

## Methods

### InitializeComponent()

InitializeComponent

```
public void InitializeComponent()
```

# Namespace MBFWpfToolkit.Controls.Help

## Classes

### [HelpWizard](#)

Represents an interactive in-app help wizard that guides users through UI elements step-by-step.

# Class HelpWizard

Namespace: [MBFWpfToolkit.Controls.Help](#)

Assembly: MBFWpfToolkit.dll

Represents an interactive in-app help wizard that guides users through UI elements step-by-step.

```
public class HelpWizard : MbWindow, INotifyPropertyChanged, IComponentConnector
```

## Inheritance

[object](#) ← [MbWindow](#) ← HelpWizard

## Implements

[INotifyPropertyChanged](#), [IComponentConnector](#)

## Inherited Members

[MbWindow.IsThemeToggleVisibleProperty](#), [MbWindow.ShowLanguagesComboBoxProperty](#),  
[MbWindow.PluginNameProperty](#), [MbWindow.PluginDescriptionProperty](#),  
[MbWindow.PluginVersionProperty](#), [MbWindow.PrimaryBrushProperty](#), [MbWindow.ShowLogoProperty](#),  
[MbWindow.ShowPluginNameProperty](#), [MbWindow.ShowPluginVersionProperty](#),  
[MbWindow.OnIsDarkChanged\(bool, bool\)](#), [MbWindow.GetAllChildrenOfType<T>\(DependencyObject\)](#),  
[MbWindow.ApplyThemeForElement\(FrameworkElement, string\)](#), [MbWindow.InitializeWindow\(\)](#),  
[MbWindow.ShowSprite\(\)](#), [MbWindow.GetPrimaryBrush\(\)](#), [MbWindow.OnPropertyChanged\(string\)](#),  
[MbWindow.SetField<T>\(ref T, T, string\)](#), [MbWindow.IsDark](#), [MbWindow.SelectedLanguage](#),  
[MbWindow.SupportedLanguages](#), [MbWindow.this\[string\]](#), [MbWindow.IsThemeToggleVisible](#),  
[MbWindow.ShowLanguagesComboBox](#), [MbWindow.PluginName](#), [MbWindow.PluginDescription](#),  
[MbWindow.PluginVersion](#), [MbWindow.PrimaryBrush](#), [MbWindow.ShowLogo](#),  
[MbWindow.ShowPluginName](#), [MbWindow.ShowPluginVersion](#), [MbWindow.MainResources](#),  
[MbWindow.IsDarkChanged](#), [MbWindow.PropertyChanged](#)

# Constructors

## HelpWizard(MbWindow)

Initializes a new instance of the [HelpWizard](#) class.

```
public HelpWizard(MbWindow mainWindow)
```

## Parameters

## `mainWindow` [MbWindow](#)

The main application window hosting the UI elements to be highlighted in the help wizard.

# Properties

## CurrentStep

Gets or sets the index of the currently active help step.

```
public int CurrentStep { get; set; }
```

Property Value

[int](#)

## Steps

Gets or sets the list of all help steps.

```
public List<HelpStep> Steps { get; set; }
```

Property Value

[List](#) <[HelpStep](#)>

# Methods

## InitializeComponent()

InitializeComponent

```
public void InitializeComponent()
```

# Namespace MBFWpfToolkit.Converters

## Classes

### [BooleanToStringConverter](#)

Converts a `bool` value to a `string` based on a semicolon-delimited parameter string.

### [BooleanErrorGroupToEnabledConverter](#)

A multi-value converter that checks a list of boolean values (typically `HasError` flags). If any value is `true`, it returns `false` to disable a control (e.g., a button).

### [BooleanToVisibilityWithInverseConverter](#)

Converts a `bool` value to a `Visibility` value. Supports optional inversion using a converter parameter.

### [EnumDisplayNameConverter](#)

A value converter for converting between enum values and their display names specified by the [`DisplayAttribute`](#).

### [NullToBooleanConverter](#)

Converts a value to `Visibility` based on whether it is null.

### [NullToVisibilityConverter](#)

Converts a value to `Visibility` based on whether it is null.

### [NullToVisibilityWithInverseConverter](#)

Converts a `null` value to a `Visibility` value. Supports optional inversion using a converter parameter.

### [ValidationMultiConverter](#)

A multi-value converter that validates a group of input values (typically bound from multiple `TextBox` elements). It checks for empty input and validation errors to determine whether a button (or any control) should be enabled.

# Class Boolean2StringConverter

Namespace: [MBFWpfToolkit.Converters](#)

Assembly: MBFWpfToolkit.dll

Converts a [bool](#) value to a [string](#) based on a semicolon-delimited parameter string.

```
public class Boolean2StringConverter : IValueConverter
```

## Inheritance

[object](#) ← Boolean2StringConverter

## Implements

[IValueConverter](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

Used to convert a boolean to two possible string values (e.g., "Off;On") for UI display.

## Constructors

### Boolean2StringConverter()

```
public Boolean2StringConverter()
```

## Methods

### Convert(object, Type, object, CultureInfo)

Converts a boolean value to a string.

```
public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
```

## Parameters

**value** [object](#)

The boolean value to convert.

**targetType** [Type](#)

The target type (should be [string](#)).

**parameter** [object](#)

A semicolon-separated string representing the two output values (e.g., "FalseValue;TrueValue").

**culture** [CultureInfo](#)

The current culture (not used).

## Returns

[object](#)

Returns the second string (after semicolon) if **value** is **true**, or the first string if **false**. Returns an empty string if input is invalid.

## ConvertBack(object, Type, object, CultureInfo)

Not implemented. Conversion back is not supported.

```
public object ConvertBack(object value, Type targetType, object parameter,  
CultureInfo culture)
```

## Parameters

**value** [object](#)

**targetType** [Type](#)

**parameter** [object](#)

**culture** [CultureInfo](#)

## Returns

[object](#)

[DoNothing](#)

# Class BooleanErrorGroupToEnabledConverter

Namespace: [MBFWpfToolkit.Converters](#)

Assembly: MBFWpfToolkit.dll

A multi-value converter that checks a list of boolean values (typically HasError flags). If any value is `true`, it returns `false` to disable a control (e.g., a button).

```
public class BooleanErrorGroupToEnabledConverter : IMultiValueConverter
```

## Inheritance

[object](#) ← BooleanErrorGroupToEnabledConverter

## Implements

[IMultiValueConverter](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### BooleanErrorGroupToEnabledConverter()

```
public BooleanErrorGroupToEnabledConverter()
```

## Methods

### Convert(object[], Type, object, CultureInfo)

Converts a group of boolean values to determine if a control should be enabled. If any value is `true`, the result is `false`.

```
public object Convert(object[] values, Type targetType, object parameter,  
CultureInfo culture)
```

## Parameters

**values** [object](#)[]

An array of boolean values indicating error states.

**targetType** [Type](#)[]

The target type of the binding (usually [bool](#)).

**parameter** [object](#)[]

Optional parameter (not used).

**culture** [CultureInfo](#)[]

The culture info (not used).

## Returns

[object](#)[]

**false** if any value is true; otherwise **true**.

## ConvertBack(object, Type[], object, CultureInfo)

ConvertBack is not implemented. Returns [DoNothing](#).

```
public object[] ConvertBack(object value, Type[] targetTypes, object parameter,  
CultureInfo culture)
```

## Parameters

**value** [object](#)[]

**targetTypes** [Type](#)[]

**parameter** [object](#)[]

**culture** [CultureInfo](#)[]

## Returns

[object](#)[]



# Class BooleanToVisibilityWithInverseConverter

Namespace: [MBFWpfToolkit.Converters](#)

Assembly: MBFWpfToolkit.dll

Converts a [bool](#) value to a [Visibility](#) value. Supports optional inversion using a converter parameter.

```
public class BooleanToVisibilityWithInverseConverter : IValueConverter
```

## Inheritance

[object](#) ← BooleanToVisibilityWithInverseConverter

## Implements

[IValueConverter](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

If the parameter is set to "Inverse" (case-insensitive), the visibility result will be inverted. Useful for showing or hiding UI elements based on boolean flags with optional inversion.

## Constructors

### BooleanToVisibilityWithInverseConverter()

```
public BooleanToVisibilityWithInverseConverter()
```

## Methods

### Convert(object, Type, object, CultureInfo)

Converts a [bool](#) to [Visibility](#).

```
public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
```

## Parameters

**value** [object](#)

The boolean value to convert.

**targetType** [Type](#)

The target type (expected to be [Visibility](#)).

**parameter** [object](#)

Optional parameter. If set to "Inverse", the result will be inverted.

**culture** [CultureInfo](#)

Culture info (not used).

## Returns

[object](#)

Returns [Visible](#) if **value** is true, or [Collapsed](#) if false. If "Inverse" is passed as a parameter, the logic is flipped.

## ConvertBack(object, Type, object, CultureInfo)

Converts a [Visibility](#) value back to [bool](#).

```
public object ConvertBack(object value, Type targetType, object parameter,  
CultureInfo culture)
```

## Parameters

**value** [object](#)

The visibility value to convert.

**targetType** [Type](#)

The target type (expected to be [bool](#)).

**parameter** [object](#)

Optional parameter (not used in ConvertBack).

**culture** [CultureInfo](#)

Culture info (not used).

Returns

**object**

Returns **true** if **value** is [Visible](#), otherwise **false**.

# Class EnumDisplayNameConverter

Namespace: [MBFWpfToolkit.Converters](#)

Assembly: MBFWpfToolkit.dll

A value converter for converting between enum values and their display names specified by the [Display Attribute](#).

```
public class EnumDisplayNameConverter : IValueConverter
```

## Inheritance

[object](#) ← EnumDisplayNameConverter

## Implements

[IValueConverter](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### EnumDisplayNameConverter()

```
public EnumDisplayNameConverter()
```

## Methods

### Convert(object, Type, object, CultureInfo)

Converts an [Enum](#) value to its display name specified in the [DisplayAttribute](#).

```
public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
```

## Parameters

### **value** [object](#)

The enum value to be converted.

### **targetType** [Type](#)

The target type of the binding (not used).

### **parameter** [object](#)

An optional parameter for the converter (not used).

### **culture** [CultureInfo](#)

The culture to be used in the converter (not used).

Returns

### [object](#)

The display name of the enum value, or its string representation if no [DisplayAttribute](#) is found.

## ConvertBack(object, Type, object, CultureInfo)

Converts a display name back to the corresponding [Enum](#) value.

```
public object ConvertBack(object value, Type targetType, object parameter,  
CultureInfo culture)
```

Parameters

### **value** [object](#)

The display name to be converted back to the enum value.

### **targetType** [Type](#)

The target enum type.

### **parameter** [object](#)

An optional parameter for the converter (not used).

### **culture** [CultureInfo](#)

The culture to be used in the converter (not used).

Returns

[object](#)

The corresponding enum value that matches the display name or field name.

Exceptions

[ArgumentException](#)

Thrown if no matching enum value is found for the display name.

# Class NullToBooleanConverter

Namespace: [MBFWpfToolkit.Converters](#)

Assembly: MBFWpfToolkit.dll

Converts a value to [Visibility](#) based on whether it is null.

```
public class NullToBooleanConverter : IValueConverter
```

## Inheritance

[object](#) ← NullToBooleanConverter

## Implements

[IValueConverter](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

If the parameter is "Inverse" (case-insensitive), visibility is inverted.

## Constructors

### NullToBooleanConverter()

```
public NullToBooleanConverter()
```

## Methods

### Convert(object?, Type, object, CultureInfo)

```
public object Convert(object? value, Type targetType, object parameter, CultureInfo culture)
```

## Parameters

`value` [object](#)

`targetType` [Type](#)

`parameter` [object](#)

`culture` [CultureInfo](#)

Returns

[object](#)

## ConvertBack(object, Type, object, CultureInfo)

```
public object ConvertBack(object value, Type targetType, object parameter,  
CultureInfo culture)
```

Parameters

`value` [object](#)

`targetType` [Type](#)

`parameter` [object](#)

`culture` [CultureInfo](#)

Returns

[object](#)

# Class NullToVisibilityConverter

Namespace: [MBFWpfToolkit.Converters](#)

Assembly: MBFWpfToolkit.dll

Converts a value to [Visibility](#) based on whether it is null.

```
public class NullToVisibilityConverter : IValueConverter
```

## Inheritance

[object](#) ← NullToVisibilityConverter

## Implements

[IValueConverter](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

If the parameter is "Inverse" (case-insensitive), visibility is inverted.

## Constructors

NullToVisibilityConverter()

```
public NullToVisibilityConverter()
```

## Methods

Convert(object, Type, object, CultureInfo)

```
public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
```

## Parameters

`value` [object](#)

`targetType` [Type](#)

`parameter` [object](#)

`culture` [CultureInfo](#)

Returns

[object](#)

## ConvertBack(object, Type, object, CultureInfo)

```
public object ConvertBack(object value, Type targetType, object parameter,  
CultureInfo culture)
```

Parameters

`value` [object](#)

`targetType` [Type](#)

`parameter` [object](#)

`culture` [CultureInfo](#)

Returns

[object](#)

# Class NullToVisibilityWithInverseConverter

Namespace: [MBFWpfToolkit.Converters](#)

Assembly: MBFWpfToolkit.dll

Converts a `null` value to a [Visibility](#) value. Supports optional inversion using a converter parameter.

```
public class NullToVisibilityWithInverseConverter : IValueConverter
```

## Inheritance

[object](#) ← NullToVisibilityWithInverseConverter

## Implements

[IValueConverter](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

If the parameter is set to "Inverse" (case-insensitive), the visibility result will be inverted. Useful for showing/hiding UI elements based on null presence.

## Constructors

### NullToVisibilityWithInverseConverter()

```
public NullToVisibilityWithInverseConverter()
```

## Methods

### Convert(object, Type, object, CultureInfo)

Converts a `null` value to [Collapsed](#) or [Visible](#).

```
public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
```

## Parameters

**value** [object](#)

The value to check for null.

**targetType** [Type](#)

The target type (expected to be [Visibility](#)).

**parameter** [object](#)

Optional parameter. If set to "Inverse", the result is inverted.

**culture** [CultureInfo](#)

Culture info (not used).

## Returns

[object](#)

[Visible](#) if the value is not null; otherwise [Collapsed](#). If "Inverse" is passed, the result is flipped.

## ConvertBack(object, Type, object, CultureInfo)

Not supported: converts back from [Visibility](#) to object.

```
public object ConvertBack(object value, Type targetType, object parameter,  
CultureInfo culture)
```

## Parameters

**value** [object](#)

**targetType** [Type](#)

**parameter** [object](#)

**culture** [CultureInfo](#)

## Returns

[object](#)



# Class ValidationMultiConverter

Namespace: [MBFWpfToolkit.Converters](#)

Assembly: MBFWpfToolkit.dll

A multi-value converter that validates a group of input values (typically bound from multiple [TextBox](#) elements). It checks for empty input and validation errors to determine whether a button (or any control) should be enabled.

```
public class ValidationMultiConverter : IMultiValueConverter
```

## Inheritance

[object](#) ← ValidationMultiConverter

## Implements

[IMultiValueConverter](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This converter expects alternating values: [Text, HasError, Text, HasError, ...]. If any Text is null/empty or any HasError is true, the result is false.

## Constructors

### ValidationMultiConverter()

```
public ValidationMultiConverter()
```

## Methods

### Convert(object[], Type, object, CultureInfo)

Validates a group of values to determine if a control (e.g., a button) should be enabled.

```
public object Convert(object[] values, Type targetType, object parameter,  
CultureInfo culture)
```

## Parameters

**values** [object](#)[]

An array of values from multiple bindings. Alternating pattern: Text, HasError, Text, HasError...

**targetType** [Type](#)

The target type of the binding (usually [bool](#)).

**parameter** [object](#)

Optional parameter (not used).

**culture** [CultureInfo](#)

The culture info (not used).

## Returns

[object](#)

Returns [true](#) if all inputs are non-empty and have no validation errors; otherwise [false](#).

## ConvertBack(object, Type[], object, CultureInfo)

ConvertBack is not implemented. Returns [DoNothing](#).

```
public object[] ConvertBack(object value, Type[] targetTypes, object parameter,  
CultureInfo culture)
```

## Parameters

**value** [object](#)

**targetTypes** [Type](#)[]

**parameter** [object](#)

culture [CultureInfo](#)

Returns

[object](#)[]

An array of [DoNothing](#).

# Namespace MBFWpfToolkit.Dialogs

## Classes

### [LoadingControl](#)

Represents a user control that displays a loading indicator.

### [LoadingDialog](#)

Represents a dialog that displays a loading indicator to the user.

### [ProgressDialog](#)

Represents a dialog that displays progress information and allows updating its message dynamically.

# Class LoadingControl

Namespace: [MBFWpfToolkit.Dialogs](#)

Assembly: MBFWpfToolkit.dll

Represents a user control that displays a loading indicator.

```
public class LoadingControl : UserControl, IAnimatable, IFrameworkInputElement,
IInputElement, ISupportInitialize, IQueryAmbient, IAddChild, IComponentConnector
```

## Inheritance

```
object ↳ ← DispatcherObject ↳ ← DependencyObject ↳ ← Visual ↳ ← UIElement ↳ ←
FrameworkElement ↳ ← Control ↳ ← ContentControl ↳ ← UserControl ↳ ← LoadingControl
```

## Implements

```
IAnimatable, IFrameworkInputElement, IInputElement, ISupportInitialize, IQueryAmbient,
IAddChild, IComponentConnector
```

## Inherited Members

```
UserControl.OnCreateAutomationPeer() ↳ , ContentControl.ContentProperty ↳ ,
ContentControl.HasContentProperty ↳ , ContentControl.ContentTemplateProperty ↳ ,
ContentControl.ContentTemplateSelectorProperty ↳ , ContentControl.ContentStringFormatProperty ↳ ,
ContentControl.AddChild(object) ↳ , ContentControl.AddText(string) ↳ ,
ContentControl.OnContentChanged(object, object) ↳ ,
ContentControl.OnContentTemplateChanged(DataTemplate, DataTemplate) ↳ ,
ContentControl.OnContentTemplateSelectorChanged(DataTemplateSelector, DataTemplateSelector) ↳ ,
ContentControl.OnContentStringFormatChanged(string, string) ↳ , ContentControl.LogicalChildren ↳ ,
ContentControl.Content ↳ , ContentControl.HasContent ↳ , ContentControl.ContentTemplate ↳ ,
ContentControl.ContentTemplateSelector ↳ , ContentControl.ContentStringFormat ↳ ,
Control.BorderBrushProperty ↳ , Control.BorderThicknessProperty ↳ , Control.BackgroundProperty ↳ ,
Control.ForegroundProperty ↳ , Control.FontFamilyProperty ↳ , Control.FontSizeProperty ↳ ,
Control.FontStretchProperty ↳ , Control.FontStyleProperty ↳ , Control.FontWeightProperty ↳ ,
Control.HorizontalContentAlignmentProperty ↳ , Control.VerticalContentAlignmentProperty ↳ ,
Control.TabIndexProperty ↳ , Control.IsTabStopProperty ↳ , Control.PaddingProperty ↳ ,
Control.TemplateProperty ↳ , Control.PreviewMouseDoubleClickEvent ↳ ,
Control.MouseDoubleClickEvent ↳ , Control.OnTemplateChanged(ControlTemplate, ControlTemplate) ↳ ,
Control.ToString() ↳ , Control.OnPreviewMouseDoubleClick(MouseEventArgs) ↳ ,
Control.OnMouseDoubleClick(MouseEventArgs) ↳ , Control.MeasureOverride(Size) ↳ ,
Control.ArrangeOverride(Size) ↳ , Control.BorderBrush ↳ , Control.BorderThickness ↳ ,
Control.Background ↳ , Control.Foreground ↳ , Control.FontFamily ↳ , Control.FontSize ↳ ,
```

[Control.FontStretch](#) , [Control.FontStyle](#) , [Control.FontWeight](#) ,  
[Control.HorizontalContentAlignment](#) , [Control.VerticalContentAlignment](#) , [Control.TabIndex](#) ,  
[Control.IsTabStop](#) , [Control.Padding](#) , [Control.Template](#) , [Control.HandlesScrolling](#) ,  
[Control.PreviewMouseDoubleClick](#) , [Control.MouseDoubleClick](#) , [FrameworkElement.StyleProperty](#) ,  
[FrameworkElement.OverridesDefaultStyleProperty](#) , [FrameworkElement.UseLayoutRoundingProperty](#) ,  
[FrameworkElement.DefaultStyleKeyProperty](#) , [FrameworkElement.DataContextProperty](#) ,  
[FrameworkElement.BindingGroupProperty](#) , [FrameworkElement.LanguageProperty](#) ,  
[FrameworkElement.NameProperty](#) , [FrameworkElement.TagProperty](#) ,  
[FrameworkElement.InputScopeProperty](#) , [FrameworkElement.RequestBringIntoViewEvent](#) ,  
[FrameworkElement.SizeChangedEvent](#) , [FrameworkElement.ActualWidthProperty](#) ,  
[FrameworkElement.ActualHeightProperty](#) , [FrameworkElement.LayoutTransformProperty](#) ,  
[FrameworkElement.WidthProperty](#) , [FrameworkElement.MinWidthProperty](#) ,  
[FrameworkElement.MaxWidthProperty](#) , [FrameworkElement.HeightProperty](#) ,  
[FrameworkElement.MinHeightProperty](#) , [FrameworkElement.MaxHeightProperty](#) ,  
[FrameworkElement.FlowDirectionProperty](#) , [FrameworkElement.MarginProperty](#) ,  
[FrameworkElement.HorizontalAlignmentProperty](#) , [FrameworkElement.VerticalAlignmentProperty](#) ,  
[FrameworkElement.FocusVisualStyleProperty](#) , [FrameworkElement.CursorProperty](#) ,  
[FrameworkElement.ForceCursorProperty](#) , [FrameworkElement.LoadedEvent](#) ,  
[FrameworkElement.UnloadedEvent](#) , [FrameworkElement.ToolTipProperty](#) ,  
[FrameworkElement.ContextMenuProperty](#) , [FrameworkElement.ToolTipOpeningEvent](#) ,  
[FrameworkElement.ToolTipClosingEvent](#) , [FrameworkElement.ContextMenuOpeningEvent](#) ,  
[FrameworkElement.ContextMenuClosingEvent](#) , [FrameworkElement.OnStyleChanged\(Style, Style\)](#) ,  
[FrameworkElement.ParentLayoutInvalidated\(UIElement\)](#) , [FrameworkElement.ApplyTemplate\(\)](#) ,  
[FrameworkElement.OnApplyTemplate\(\)](#) , [FrameworkElement.BeginStoryboard\(Storyboard\)](#) ,  
[FrameworkElement.BeginStoryboard\(Storyboard, HandoffBehavior\)](#) ,  
[FrameworkElement.BeginStoryboard\(Storyboard, HandoffBehavior, bool\)](#) ,  
[FrameworkElement.GetVisualChild\(int\)](#) , [FrameworkElement.GetTemplateChild\(string\)](#) ,  
[FrameworkElement.FindResource\(object\)](#) , [FrameworkElement.TryFindResource\(object\)](#) ,  
[FrameworkElement.SetResourceReference\(DependencyProperty, object\)](#) ,  
[FrameworkElement.OnPropertyChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[FrameworkElement.OnVisualParentChanged\(DependencyObject\)](#) ,  
[FrameworkElement.GetBindingExpression\(DependencyProperty\)](#) ,  
[FrameworkElement.SetBinding\(DependencyProperty, BindingBase\)](#) ,  
[FrameworkElement.SetBinding\(DependencyProperty, string\)](#) , [FrameworkElement.GetUIParentCore\(\)](#) ,  
[FrameworkElement.BringIntoView\(\)](#) , [FrameworkElement.BringIntoView\(Rect\)](#) ,  
[FrameworkElement.GetFlowDirection\(DependencyObject\)](#) ,  
[FrameworkElement.SetFlowDirection\(DependencyObject, FlowDirection\)](#) ,  
[FrameworkElement.MeasureCore\(Size\)](#) , [FrameworkElement.ArrangeCore\(Rect\)](#) ,  
[FrameworkElement.OnRenderSizeChanged\(SizeChangedEventArgs\)](#) ,  
[FrameworkElement.GetLayoutClip\(Size\)](#) , [FrameworkElement.MoveFocus\(TraversalRequest\)](#) ,

[FrameworkElement.PredictFocus\(FocusNavigationDirection\)](#) ,  
[FrameworkElement.OnGotFocus\(RoutedEventArgs\)](#) , [FrameworkElement.BeginInit\(\)](#) ,  
[FrameworkElement.EndInit\(\)](#) , [FrameworkElement.OnInitialized\(EventArgs\)](#) ,  
[FrameworkElement.OnToolTipOpening\(ToolTipEventArgs\)](#) ,  
[FrameworkElement.OnToolTipClosing\(ToolTipEventArgs\)](#) ,  
[FrameworkElement.OnContextMenuOpening\(ContextMenuEventArgs\)](#) ,  
[FrameworkElement.OnContextMenuClosing\(ContextMenuEventArgs\)](#) ,  
[FrameworkElement.RegisterName\(string, object\)](#) , [FrameworkElement.UnregisterName\(string\)](#) ,  
[FrameworkElement.FindName\(string\)](#) , [FrameworkElement.UpdateDefaultStyle\(\)](#) ,  
[FrameworkElement.AddLogicalChild\(object\)](#) , [FrameworkElement.RemoveLogicalChild\(object\)](#) ,  
[FrameworkElement.Style](#) , [FrameworkElement.OverridesDefaultStyle](#) ,  
[FrameworkElement.UseLayoutRounding](#) , [FrameworkElement.DefaultStyleKey](#) ,  
[FrameworkElement.Triggers](#) , [FrameworkElement.TemplatedParent](#) ,  
[FrameworkElement.VisualChildrenCount](#) , [FrameworkElement.Resources](#) ,  
[FrameworkElement.InheritanceBehavior](#) , [FrameworkElement.DataContext](#) ,  
[FrameworkElement.BindingGroup](#) , [FrameworkElement.Language](#) , [FrameworkElement.Name](#) ,  
[FrameworkElement.Tag](#) , [FrameworkElement.InputScope](#) , [FrameworkElement.ActualWidth](#) ,  
[FrameworkElement.ActualHeight](#) , [FrameworkElement.LayoutTransform](#) ,  
[FrameworkElement.Width](#) , [FrameworkElement.MinWidth](#) , [FrameworkElement.MaxWidth](#) ,  
[FrameworkElement.Height](#) , [FrameworkElement.MinHeight](#) , [FrameworkElement.MaxHeight](#) ,  
[FrameworkElement.FlowDirection](#) , [FrameworkElement.Margin](#) ,  
[FrameworkElement.HorizontalAlignment](#) , [FrameworkElement.VerticalAlignment](#) ,  
[FrameworkElement.FocusVisualStyle](#) , [FrameworkElement.Cursor](#) , [FrameworkElement.ForceCursor](#) ,  
[FrameworkElement.IsInitialized](#) , [FrameworkElement.IsLoaded](#) , [FrameworkElement.ToolTip](#) ,  
[FrameworkElement.ContextMenu](#) , [FrameworkElement.Parent](#) , [FrameworkElement.TargetUpdated](#) ,  
[FrameworkElement.SourceUpdated](#) , [FrameworkElement.DataContextChanged](#) ,  
[FrameworkElement.RequestBringIntoView](#) , [FrameworkElement.SizeChanged](#) ,  
[FrameworkElement.Initialized](#) , [FrameworkElement.Loaded](#) , [FrameworkElement.Unloaded](#) ,  
[FrameworkElement.ToolTipOpening](#) , [FrameworkElement.ToolTipClosing](#) ,  
[FrameworkElement.ContextMenuOpening](#) , [FrameworkElement.ContextMenuClosing](#) ,  
[UIElement.PreviewMouseDownEvent](#) , [UIElement.MouseDownEvent](#) ,  
[UIElement.PreviewMouseUpEvent](#) , [UIElement.MouseUpEvent](#) ,  
[UIElement.PreviewMouseLeftButtonDownEvent](#) , [UIElement.MouseLeftButtonDownEvent](#) ,  
[UIElement.PreviewMouseLeftButtonUpEvent](#) , [UIElement.MouseLeftButtonUpEvent](#) ,  
[UIElement.PreviewMouseRightButtonDownEvent](#) , [UIElement.MouseRightButtonDownEvent](#) ,  
[UIElement.PreviewMouseRightButtonUpEvent](#) , [UIElement.MouseRightButtonUpEvent](#) ,  
[UIElement.PreviewMouseMoveEvent](#) , [UIElement.MouseMoveEvent](#) ,  
[UIElement.PreviewMouseWheelEvent](#) , [UIElement.MouseWheelEvent](#) , [UIElement.MouseEnterEvent](#) ,  
[UIElement.MouseLeaveEvent](#) , [UIElement.GotMouseCaptureEvent](#) ,  
[UIElement.LostMouseCaptureEvent](#) , [UIElement.QueryCursorEvent](#) ,

[UIElement.PreviewStylusDownEvent](#) , [UIElement.StylusDownEvent](#) ,  
[UIElement.PreviewStylusUpEvent](#) , [UIElement.StylusUpEvent](#) , [UIElement.PreviewStylusMoveEvent](#) ,  
[UIElement.StylusMoveEvent](#) , [UIElement.PreviewStylusInAirMoveEvent](#) ,  
[UIElement.StylusInAirMoveEvent](#) , [UIElement.StylusEnterEvent](#) , [UIElement.StylusLeaveEvent](#) ,  
[UIElement.PreviewStylusInRangeEvent](#) , [UIElement.StylusInRangeEvent](#) ,  
[UIElement.PreviewStylusOutOfRangeEvent](#) , [UIElement.StylusOutOfRangeEvent](#) ,  
[UIElement.PreviewStylusSystemGestureEvent](#) , [UIElement.StylusSystemGestureEvent](#) ,  
[UIElement.GotStylusCaptureEvent](#) , [UIElement.LostStylusCaptureEvent](#) ,  
[UIElement.StylusButtonDownEvent](#) , [UIElement.StylusButtonUpEvent](#) ,  
[UIElement.PreviewStylusButtonDownEvent](#) , [UIElement.PreviewStylusButtonUpEvent](#) ,  
[UIElement.PreviewKeyDownEvent](#) , [UIElement.KeyDownEvent](#) , [UIElement.PreviewKeyUpEvent](#) ,  
[UIElement.KeyUpEvent](#) , [UIElement.PreviewGotKeyboardFocusEvent](#) ,  
[UIElement.GotKeyboardFocusEvent](#) , [UIElement.PreviewLostKeyboardFocusEvent](#) ,  
[UIElement.LostKeyboardFocusEvent](#) , [UIElement.PreviewTextInputEvent](#) , [UIElement.TextInputEvent](#) ,  
[UIElement.PreviewQueryContinueDragEvent](#) , [UIElement.QueryContinueDragEvent](#) ,  
[UIElement.PreviewGiveFeedbackEvent](#) , [UIElement.GiveFeedbackEvent](#) ,  
[UIElement.PreviewDragEnterEvent](#) , [UIElement.DragEnterEvent](#) , [UIElement.PreviewDragOverEvent](#) ,  
[UIElement.DragOverEvent](#) , [UIElement.PreviewDragLeaveEvent](#) , [UIElement.DragLeaveEvent](#) ,  
[UIElement.PreviewDropEvent](#) , [UIElement.DropEvent](#) , [UIElement.PreviewTouchDownEvent](#) ,  
[UIElement.TouchDownEvent](#) , [UIElement.PreviewTouchMoveEvent](#) , [UIElement.TouchMoveEvent](#) ,  
[UIElement.PreviewTouchUpEvent](#) , [UIElement.TouchUpEvent](#) , [UIElement.GotTouchCaptureEvent](#) ,  
[UIElement.LostTouchCaptureEvent](#) , [UIElement.TouchEnterEvent](#) , [UIElement.TouchLeaveEvent](#) ,  
[UIElement.IsMouseDirectlyOverProperty](#) , [UIElement.IsMouseOverProperty](#) ,  
[UIElement.IsStylusOverProperty](#) , [UIElement.IsKeyboardFocusWithinProperty](#) ,  
[UIElement.IsMouseCapturedProperty](#) , [UIElement.IsMouseCaptureWithinProperty](#) ,  
[UIElement.IsStylusDirectlyOverProperty](#) , [UIElement.IsStylusCapturedProperty](#) ,  
[UIElement.IsStylusCaptureWithinProperty](#) , [UIElement.IsKeyboardFocusedProperty](#) ,  
[UIElement.AreAnyTouchesDirectlyOverProperty](#) , [UIElement.AreAnyTouchesOverProperty](#) ,  
[UIElement.AreAnyTouchesCapturedProperty](#) , [UIElement.AreAnyTouchesCapturedWithinProperty](#) ,  
[UIElement.AllowDropProperty](#) , [UIElement.RenderTransformProperty](#) ,  
[UIElement.RenderTransformOriginProperty](#) , [UIElement.OpacityProperty](#) ,  
[UIElement.OpacityMaskProperty](#) , [UIElement.BitmapEffectProperty](#) , [UIElement.EffectProperty](#) ,  
[UIElement.BitmapEffectInputProperty](#) , [UIElement.CacheModeProperty](#) , [UIElement.UidProperty](#) ,  
[UIElement.VisibilityProperty](#) , [UIElement.ClipToBoundsProperty](#) , [UIElement.ClipProperty](#) ,  
[UIElement.SnapsToDevicePixelsProperty](#) , [UIElement.GotFocusEvent](#) , [UIElement.LostFocusEvent](#) ,  
[UIElement.FocusedProperty](#) , [UIElement.IsEnabledProperty](#) , [UIElement.IsHitTestVisibleProperty](#) ,  
[UIElement.VisibleProperty](#) , [UIElement.FocusableProperty](#) ,  
[UIElement.ManipulationEnabledProperty](#) , [UIElement.ManipulationStartingEvent](#) ,  
[UIElement.ManipulationStartedEvent](#) , [UIElement.ManipulationDeltaEvent](#) ,  
[UIElement.ManipulationInertiaStartingEvent](#) , [UIElement.ManipulationBoundaryFeedbackEvent](#) ,

[UIElement.ManipulationCompletedEvent](#) ,  
[UIElement.ApplyAnimationClock\(DependencyProperty, AnimationClock\)](#) ,  
[UIElement.ApplyAnimationClock\(DependencyProperty, AnimationClock, HandoffBehavior\)](#) ,  
[UIElement.BeginAnimation\(DependencyProperty, AnimationTimeline\)](#) ,  
[UIElement.BeginAnimation\(DependencyProperty, AnimationTimeline, HandoffBehavior\)](#) ,  
[UIElement.GetAnimationBaseValue\(DependencyProperty\)](#) , [UIElement.RaiseEvent\(RoutedEventArgs\)](#) ,  
[UIElement.AddHandler\(RoutedEvent, Delegate\)](#) ,  
[UIElement.AddHandler\(RoutedEvent, Delegate, bool\)](#) ,  
[UIElement.RemoveHandler\(RoutedEvent, Delegate\)](#) ,  
[UIElement.AddToEventRoute\(EventRoute, RoutedEventArgs\)](#) ,  
[UIElement.OnPreviewMouseDown\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseDown\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseUp\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseUp\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseLeftButtonDown\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseLeftButtonDown\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseLeftButtonUp\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseLeftButtonUp\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseRightButtonDown\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseRightButtonDown\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseRightButtonUp\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseRightButtonUp\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseMove\(MouseEventArgs\)](#) , [UIElement.OnMouseMove\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseWheel\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseWheel\(MouseEventArgs\)](#) , [UIElement.OnMouseEnter\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseLeave\(MouseEventArgs\)](#) , [UIElement.OnGotMouseCapture\(MouseEventArgs\)](#) ,  
[UIElement.OnLostMouseCapture\(MouseEventArgs\)](#) ,  
[UIElement.OnQueryCursor\(QueryEventArgs\)](#) ,  
[UIElement.OnPreviewStylusDown\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusDown\(StylusEventArgs\)](#) , [UIElement.OnPreviewStylusUp\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusUp\(StylusEventArgs\)](#) , [UIElement.OnPreviewStylusMove\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusMove\(StylusEventArgs\)](#) , [UIElement.OnPreviewStylusInAirMove\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusInAirMove\(StylusEventArgs\)](#) , [UIElement.OnStylusEnter\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusLeave\(StylusEventArgs\)](#) , [UIElement.OnPreviewStylusInRange\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusInRange\(StylusEventArgs\)](#) ,  
[UIElement.OnPreviewStylusOutOfRange\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusOutOfRange\(StylusEventArgs\)](#) ,  
[UIElement.OnPreviewStylusSystemGesture\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusSystemGesture\(StylusEventArgs\)](#) ,  
[UIElement.OnGotStylusCapture\(StylusEventArgs\)](#) , [UIElement.OnLostStylusCapture\(StylusEventArgs\)](#) ,

[UIElement.OnStylusButtonDown\(StylusButtonEventArgs\)](#) ,  
[UIElement.OnStylusButtonUp\(StylusButtonEventArgs\)](#) ,  
[UIElement.OnPreviewStylusButtonDown\(StylusButtonEventArgs\)](#) ,  
[UIElement.OnPreviewStylusButtonUp\(StylusButtonEventArgs\)](#) ,  
[UIElement.OnPreviewKeyDown\(KeyEventArgs\)](#) , [UIElement.OnKeyDown\(KeyEventArgs\)](#) ,  
[UIElement.OnPreviewKeyUp\(KeyEventArgs\)](#) , [UIElement.OnKeyUp\(KeyEventArgs\)](#) ,  
[UIElement.OnPreviewGotKeyboardFocus\(KeyboardFocusChangedEventArgs\)](#) ,  
[UIElement.OnGotKeyboardFocus\(KeyboardFocusChangedEventArgs\)](#) ,  
[UIElement.OnPreviewLostKeyboardFocus\(KeyboardFocusChangedEventArgs\)](#) ,  
[UIElement.OnLostKeyboardFocus\(KeyboardFocusChangedEventArgs\)](#) ,  
[UIElement.OnPreviewTextInput\(TextCompositionEventArgs\)](#) ,  
[UIElement.OnTextInput\(TextCompositionEventArgs\)](#) ,  
[UIElement.OnPreviewQueryContinueDrag\(QueryContinueDragEventArgs\)](#) ,  
[UIElement.OnQueryContinueDrag\(QueryContinueDragEventArgs\)](#) ,  
[UIElement.OnPreviewGiveFeedback\(GiveFeedbackEventArgs\)](#) ,  
[UIElement.OnGiveFeedback\(GiveFeedbackEventArgs\)](#) ,  
[UIElement.OnPreviewDragEnter\(DragEventArgs\)](#) , [UIElement.OnDragEnter\(DragEventArgs\)](#) ,  
[UIElement.OnPreviewDragOver\(DragEventArgs\)](#) , [UIElement.OnDragOver\(DragEventArgs\)](#) ,  
[UIElement.OnPreviewDragLeave\(DragEventArgs\)](#) , [UIElement.OnDragLeave\(DragEventArgs\)](#) ,  
[UIElement.OnPreviewDrop\(DragEventArgs\)](#) , [UIElement.OnDrop\(DragEventArgs\)](#) ,  
[UIElement.OnPreviewTouchDown\(TouchEventArgs\)](#) , [UIElement.OnTouchDown\(TouchEventArgs\)](#) ,  
[UIElement.OnPreviewTouchMove\(TouchEventArgs\)](#) , [UIElement.OnTouchMove\(TouchEventArgs\)](#) ,  
[UIElement.OnPreviewTouchUp\(TouchEventArgs\)](#) , [UIElement.OnTouchUp\(TouchEventArgs\)](#) ,  
[UIElement.OnGotTouchCapture\(TouchEventArgs\)](#) , [UIElement.OnLostTouchCapture\(TouchEventArgs\)](#) ,  
[UIElement.OnTouchEnter\(TouchEventArgs\)](#) , [UIElement.OnTouchLeave\(TouchEventArgs\)](#) ,  
[UIElement.OnIsMouseDirectlyOverChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsKeyboardFocusWithinChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsMouseCapturedChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsMouseCaptureWithinChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsStylusDirectlyOverChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsStylusCapturedChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsStylusCaptureWithinChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsKeyboardFocusedChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.InvalidateMeasure\(\)](#) , [UIElement.InvalidateArrange\(\)](#) , [UIElement.InvalidateVisual\(\)](#) ,  
[UIElement.OnChildDesiredSizeChanged\(UIElement\)](#) , [UIElement.Measure\(Size\)](#) ,  
[UIElement.Arrange\(Rect\)](#) , [UIElement.OnRender\(DrawingContext\)](#) , [UIElement.UpdateLayout\(\)](#) ,  
[UIElement.TranslatePoint\(Point, UIElement\)](#) , [UIElement.InputHitTest\(Point\)](#) ,  
[UIElement.CaptureMouse\(\)](#) , [UIElement.ReleaseMouseCapture\(\)](#) , [UIElement.CaptureStylus\(\)](#) ,  
[UIElement.ReleaseStylusCapture\(\)](#) , [UIElement.Focus\(\)](#) ,  
[UIElement.OnAccessKey\(AccessKeyEventArgs\)](#) , [UIElement.HitTestCore\(PointHitTestParameters\)](#) ,

[UIElement.HitTestCore\(GeometryHitTestParameters\)](#) , [UIElement.OnLostFocus\(RoutedEventArgs\)](#) ,  
[UIElement.OnManipulationStarting\(ManipulationStartingEventArgs\)](#) ,  
[UIElement.OnManipulationStarted\(ManipulationStartedEventArgs\)](#) ,  
[UIElement.OnManipulationDelta\(ManipulationDeltaEventArgs\)](#) ,  
[UIElement.OnManipulationInertiaStarting\(ManipulationInertiaStartingEventArgs\)](#) ,  
[UIElement.OnManipulationBoundaryFeedback\(ManipulationBoundaryFeedbackEventArgs\)](#) ,  
[UIElement.OnManipulationCompleted\(ManipulationCompletedEventArgs\)](#) ,  
[UIElement.CaptureTouch\(TouchDevice\)](#) , [UIElement.ReleaseTouchCapture\(TouchDevice\)](#) ,  
[UIElement.ReleaseAllTouchCaptures\(\)](#) , [UIElement.HasAnimatedProperties](#) ,  
[UIElement.InputBindings](#) , [UIElement.CommandBindings](#) , [UIElement.AllowDrop](#) ,  
[UIElement.StylusPlugIns](#) , [UIElement.DesiredSize](#) , [UIElement.IsMeasureValid](#) ,  
[UIElement.IsArrangeValid](#) , [UIElement.RenderSize](#) , [UIElement.RenderTransform](#) ,  
[UIElement.RenderTransformOrigin](#) , [UIElement.IsMouseDirectlyOver](#) , [UIElement.IsMouseOver](#) ,  
[UIElement.IsStylusOver](#) , [UIElement.IsKeyboardFocusWithin](#) , [UIElement.IsMouseCaptured](#) ,  
[UIElement.IsMouseCaptureWithin](#) , [UIElement.IsStylusDirectlyOver](#) , [UIElement.IsStylusCaptured](#) ,  
[UIElement.IsStylusCaptureWithin](#) , [UIElement.IsKeyboardFocused](#) ,  
[UIElement.IsInputMethodEnabled](#) , [UIElement.Opacity](#) , [UIElement.OpacityMask](#) ,  
[UIElement.BitmapEffect](#) , [UIElement.Effect](#) , [UIElement.BitmapEffectInput](#) ,  
[UIElement.CacheMode](#) , [UIElement.Uid](#) , [UIElement.Visibility](#) , [UIElement.ClipToBounds](#) ,  
[UIElement.Clip](#) , [UIElement.SnapsToDevicePixels](#) , [UIElement.HasEffectiveKeyboardFocus](#) ,  
[UIElement.Focused](#) , [UIElement.IsEnabled](#) , [UIElement.IsEnabledCore](#) ,  
[UIElement.HitTestVisible](#) , [UIElement.Visible](#) , [UIElement.Focusable](#) , [UIElement.PersistId](#) ,  
[UIElement.ManipulationEnabled](#) , [UIElement.AreAnyTouchesOver](#) ,  
[UIElement.AreAnyTouchesDirectlyOver](#) , [UIElement.AreAnyTouchesCapturedWithin](#) ,  
[UIElement.AreAnyTouchesCaptured](#) , [UIElement.TouchesCaptured](#) ,  
[UIElement.TouchesCapturedWithin](#) , [UIElement.TouchesOver](#) , [UIElement.TouchesDirectlyOver](#) ,  
[UIElement.PreviewMouseDown](#) , [UIElement.MouseDown](#) , [UIElement.PreviewMouseUp](#) ,  
[UIElement.MouseUp](#) , [UIElement.PreviewMouseLeftButtonDown](#) ,  
[UIElement.MouseLeftButtonDown](#) , [UIElement.PreviewMouseLeftButtonUp](#) ,  
[UIElement.MouseLeftButtonUp](#) , [UIElement.PreviewMouseRightButtonDown](#) ,  
[UIElement.MouseRightButtonDown](#) , [UIElement.PreviewMouseRightButtonUp](#) ,  
[UIElement.MouseRightButtonUp](#) , [UIElement.PreviewMouseMove](#) , [UIElement.MouseMove](#) ,  
[UIElement.PreviewMouseWheel](#) , [UIElement.MouseWheel](#) , [UIElement.MouseEnter](#) ,  
[UIElement.MouseLeave](#) , [UIElement.GotMouseCapture](#) , [UIElement.LostMouseCapture](#) ,  
[UIElement.QueryCursor](#) , [UIElement.PreviewStylusDown](#) , [UIElement.StylusDown](#) ,  
[UIElement.PreviewStylusUp](#) , [UIElement.StylusUp](#) , [UIElement.PreviewStylusMove](#) ,  
[UIElement.StylusMove](#) , [UIElement.PreviewStylusInAirMove](#) , [UIElement.StylusInAirMove](#) ,  
[UIElement.StylusEnter](#) , [UIElement.StylusLeave](#) , [UIElement.PreviewStylusInRange](#) ,  
[UIElement.StylusInRange](#) , [UIElement.PreviewStylusOutOfRange](#) , [UIElement.StylusOutOfRange](#) ,  
[UIElement.PreviewStylusSystemGesture](#) , [UIElement.StylusSystemGesture](#) ,

[UIElement.GotStylusCapture](#) , [UIElement.LostStylusCapture](#) , [UIElement.StylusButtonDown](#) ,  
[UIElement.StylusButtonUp](#) , [UIElement.PreviewStylusButtonDown](#) ,  
[UIElement.PreviewStylusButtonUp](#) , [UIElement.PreviewKeyDown](#) , [UIElement.KeyDown](#) ,  
[UIElement.PreviewKeyUp](#) , [UIElement.KeyUp](#) , [UIElement.PreviewGotKeyboardFocus](#) ,  
[UIElement.GotKeyboardFocus](#) , [UIElement.PreviewLostKeyboardFocus](#) ,  
[UIElement.LostKeyboardFocus](#) , [UIElement.PreviewTextInput](#) , [UIElement.TextInput](#) ,  
[UIElement.PreviewQueryContinueDrag](#) , [UIElement.QueryContinueDrag](#) ,  
[UIElement.PreviewGiveFeedback](#) , [UIElement.GiveFeedback](#) , [UIElement.PreviewDragEnter](#) ,  
[UIElement.DragEnter](#) , [UIElement.PreviewDragOver](#) , [UIElement.DragOver](#) ,  
[UIElement.PreviewDragLeave](#) , [UIElement.DragLeave](#) , [UIElement.PreviewDrop](#) , [UIElement.Drop](#) ,  
[UIElement.PreviewTouchDown](#) , [UIElement.TouchDown](#) , [UIElement.PreviewTouchMove](#) ,  
[UIElement.TouchMove](#) , [UIElement.PreviewTouchUp](#) , [UIElement.TouchUp](#) ,  
[UIElement.GotTouchCapture](#) , [UIElement.LostTouchCapture](#) , [UIElement.TouchEnter](#) ,  
[UIElement.TouchLeave](#) , [UIElement.IsMouseDirectlyOverChanged](#) ,  
[UIElement.IsKeyboardFocusWithinChanged](#) , [UIElement.IsMouseCapturedChanged](#) ,  
[UIElement.IsMouseCaptureWithinChanged](#) , [UIElement.IsStylusDirectlyOverChanged](#) ,  
[UIElement.IsStylusCapturedChanged](#) , [UIElement.IsStylusCaptureWithinChanged](#) ,  
[UIElement.IsKeyboardFocusedChanged](#) , [UIElement.LayoutUpdated](#) , [UIElement.GotFocus](#) ,  
[UIElement.LostFocus](#) , [UIElement.IsEnabledChanged](#) , [UIElement.IsHitTestVisibleChanged](#) ,  
[UIElement.IsVisibleChanged](#) , [UIElement.FocusableChanged](#) , [UIElement.ManipulationStarting](#) ,  
[UIElement.ManipulationStarted](#) , [UIElement.ManipulationDelta](#) ,  
[UIElement.ManipulationInertiaStarting](#) , [UIElement.ManipulationBoundaryFeedback](#) ,  
[UIElement.ManipulationCompleted](#) , [Visual.AddVisualChild\(Visual\)](#) ,  
[Visual.RemoveVisualChild\(Visual\)](#) ,  
[Visual.OnVisualChildrenChanged\(DependencyObject, DependencyObject\)](#) ,  
[Visual.OnDpiChanged\(DpiScale, DpiScale\)](#) , [Visual.IsAncestorOf\(DependencyObject\)](#) ,  
[Visual.IsDescendantOf\(DependencyObject\)](#) , [Visual.FindCommonVisualAncestor\(DependencyObject\)](#) ,  
[Visual.TransformToAncestor\(Visual\)](#) , [Visual.TransformToAncestor\(Visual3D\)](#) ,  
[Visual.TransformToDescendant\(Visual\)](#) , [Visual.TransformToVisual\(Visual\)](#) ,  
[Visual.PointToScreen\(Point\)](#) , [Visual.PointFromScreen\(Point\)](#) , [Visual.VisualParent](#) ,  
[Visual.VisualTransform](#) , [Visual.VisualEffect](#) , [Visual.VisualBitmapEffect](#) ,  
[Visual.VisualBitmapEffectInput](#) , [Visual.VisualCacheMode](#) , [Visual.VisualScrollableAreaClip](#) ,  
[Visual.VisualClip](#) , [Visual.VisualOffset](#) , [Visual.VisualOpacity](#) , [Visual.VisualEdgeMode](#) ,  
[Visual.VisualBitmapScalingMode](#) , [Visual.VisualClearTypeHint](#) , [Visual.VisualTextRenderingMode](#) ,  
[Visual.VisualTextHintingMode](#) , [Visual.VisualOpacityMask](#) , [Visual.VisualXSnappingGuidelines](#) ,  
[Visual.VisualYSnappingGuidelines](#) , [DependencyObject.Equals\(object\)](#) ,  
[DependencyObject.GetHashCode\(\)](#) , [DependencyObject.GetValue\(DependencyProperty\)](#) ,  
[DependencyObject.SetValue\(DependencyProperty, object\)](#) ,  
[DependencyObject.SetCurrentValue\(DependencyProperty, object\)](#) ,  
[DependencyObject.SetValue\(DependencyPropertyKey, object\)](#) ,

[DependencyObject.ClearValue\(DependencyProperty\)](#) ,  
[DependencyObject.ClearValue\(DependencyPropertyKey\)](#) ,  
[DependencyObject.CoerceValue\(DependencyProperty\)](#) ,  
[DependencyObject.InvalidateProperty\(DependencyProperty\)](#) ,  
[DependencyObject.ShouldSerializeProperty\(DependencyProperty\)](#) ,  
[DependencyObject.ReadLocalValue\(DependencyProperty\)](#) ,  
[DependencyObject.GetLocalValueEnumerator\(\)](#) , [DependencyObject.DependencyObjectType](#) ,  
[DependencyObject.IsSealed](#) , [DispatcherObject.Dispatcher](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Remarks

This control is typically used to indicate that a background operation is in progress. It can be added to a user interface to provide visual feedback to the user during long-running tasks.

## Constructors

### LoadingControl()

Initializes a new instance of the [LoadingControl](#) class.

```
public LoadingControl()
```

## Remarks

This constructor sets up the [LoadingControl](#) by initializing its components. Use this control to display a loading indicator in your application.

## Fields

### LoaderTextProperty

```
public static readonly DependencyProperty LoaderTextProperty
```

## Field Value

[DependencyProperty](#)

# Properties

## LoaderText

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

## Property Value

[string](#) ↗

# Methods

## InitializeComponent()

### InitializeComponent

```
public void InitializeComponent()
```

# Class LoadingDialog

Namespace: [MBFWpfToolkit.Dialogs](#)

Assembly: MBFWpfToolkit.dll

Represents a dialog that displays a loading indicator to the user.

```
public class LoadingDialog : Border, IAnimatable, IFrameworkInputElement, IInputElement,  
ISupportInitialize, IQueryAmbient, IAddChild, IComponentConnector
```

## Inheritance

```
object ↗ ← DispatcherObject ↗ ← DependencyObject ↗ ← Visual ↗ ← UIElement ↗ ←  
FrameworkElement ↗ ← Decorator ↗ ← Border ↗ ← LoadingDialog
```

## Implements

```
IAnimatable ↗ , IFrameworkInputElement ↗ , IInputElement ↗ , ISupportInitialize ↗ , IQueryAmbient ↗ ,  
IAddChild ↗ , IComponentConnector ↗
```

## Inherited Members

```
Border.BorderThicknessProperty ↗ , Border.PaddingProperty ↗ , Border.CornerRadiusProperty ↗ ,  
Border.BorderBrushProperty ↗ , Border.BackgroundProperty ↗ , Border.MeasureOverride(Size) ↗ ,  
Border.ArrangeOverride(Size) ↗ , Border.OnRender(DrawingContext) ↗ , Border.BorderThickness ↗ ,  
Border.Padding ↗ , Border.CornerRadius ↗ , Border.BorderBrush ↗ , Border.Background ↗ ,  
Decorator.GetVisualChild(int) ↗ , Decorator.Child ↗ , Decorator.LogicalChildren ↗ ,  
Decorator.VisualChildrenCount ↗ , FrameworkElement.StyleProperty ↗ ,  
FrameworkElementOverridesDefaultStyleProperty ↗ , FrameworkElement.UseLayoutRoundingProperty ↗ ,  
FrameworkElement.DefaultStyleKeyProperty ↗ , FrameworkElement.DataContextProperty ↗ ,  
FrameworkElement.BindingGroupProperty ↗ , FrameworkElement.LanguageProperty ↗ ,  
FrameworkElement.NameProperty ↗ , FrameworkElement.TagProperty ↗ ,  
FrameworkElement.InputScopeProperty ↗ , FrameworkElement.RequestBringIntoViewEvent ↗ ,  
FrameworkElement.SizeChangedEvent ↗ , FrameworkElement.ActualWidthProperty ↗ ,  
FrameworkElement.ActualHeightProperty ↗ , FrameworkElement.LayoutTransformProperty ↗ ,  
FrameworkElement.WidthProperty ↗ , FrameworkElement.MinWidthProperty ↗ ,  
FrameworkElement.MaxWidthProperty ↗ , FrameworkElement.HeightProperty ↗ ,  
FrameworkElement.MinHeightProperty ↗ , FrameworkElement.MaxHeightProperty ↗ ,  
FrameworkElement.FlowDirectionProperty ↗ , FrameworkElement.MarginProperty ↗ ,  
FrameworkElement.HorizontalContentAlignmentProperty ↗ , FrameworkElement.VerticalAlignmentProperty ↗ ,  
FrameworkElement.FocusVisualStyleProperty ↗ , FrameworkElement.CursorProperty ↗ ,  
FrameworkElement.ForceCursorProperty ↗ , FrameworkElement.LoadedEvent ↗ ,  
FrameworkElement.UnloadedEvent ↗ , FrameworkElement.ToolTipProperty ↗ ,
```

[FrameworkElement.ContextMenuProperty](#) , [FrameworkElement.ToolTipOpeningEvent](#) ,  
[FrameworkElement.ToolTipClosingEvent](#) , [FrameworkElement.ContextMenuOpeningEvent](#) ,  
[FrameworkElement.ContextMenuClosingEvent](#) , [FrameworkElement.OnStyleChanged\(Style, Style\)](#) ,  
[FrameworkElement.ParentLayoutInvalidated\(UIElement\)](#) , [FrameworkElement.ApplyTemplate\(\)](#) ,  
[FrameworkElement.OnApplyTemplate\(\)](#) , [FrameworkElement.BeginStoryboard\(Storyboard\)](#) ,  
[FrameworkElement.BeginStoryboard\(Storyboard, HandoffBehavior\)](#) ,  
[FrameworkElement.BeginStoryboard\(Storyboard, HandoffBehavior, bool\)](#) ,  
[FrameworkElement.GetTemplateChild\(string\)](#) , [FrameworkElement.FindResource\(object\)](#) ,  
[FrameworkElement.TryFindResource\(object\)](#) ,  
[FrameworkElement.SetResourceReference\(DependencyProperty, object\)](#) ,  
[FrameworkElement.OnPropertyChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[FrameworkElement.OnVisualParentChanged\(DependencyObject\)](#) ,  
[FrameworkElement.GetBindingExpression\(DependencyProperty\)](#) ,  
[FrameworkElement.SetBinding\(DependencyProperty, BindingBase\)](#) ,  
[FrameworkElement.SetBinding\(DependencyProperty, string\)](#) , [FrameworkElement.GetUIParentCore\(\)](#) ,  
[FrameworkElement.BringIntoView\(\)](#) , [FrameworkElement.BringIntoView\(Rect\)](#) ,  
[FrameworkElement.GetFlowDirection\(DependencyObject\)](#) ,  
[FrameworkElement.SetFlowDirection\(DependencyObject, FlowDirection\)](#) ,  
[FrameworkElement.MeasureCore\(Size\)](#) , [FrameworkElement.ArrangeCore\(Rect\)](#) ,  
[FrameworkElement.OnRenderSizeChanged\(SizeChangedEventArgs\)](#) ,  
[FrameworkElement.GetLayoutClip\(Size\)](#) , [FrameworkElement.MoveFocus\(TraversalRequest\)](#) ,  
[FrameworkElement.PredictFocus\(FocusNavigationDirection\)](#) ,  
[FrameworkElement.OnGotFocus\(RoutedEventArgs\)](#) , [FrameworkElement.BeginInit\(\)](#) ,  
[FrameworkElement.EndInit\(\)](#) , [FrameworkElement.OnInitialized\(EventArgs\)](#) ,  
[FrameworkElement.OnToolTipOpening\(ToolTipEventArgs\)](#) ,  
[FrameworkElement.OnToolTipClosing\(ToolTipEventArgs\)](#) ,  
[FrameworkElement.OnContextMenuOpening\(ContextMenuEventArgs\)](#) ,  
[FrameworkElement.OnContextMenuClosing\(ContextMenuEventArgs\)](#) ,  
[FrameworkElement.RegisterName\(string, object\)](#) , [FrameworkElement.UnregisterName\(string\)](#) ,  
[FrameworkElement.FindName\(string\)](#) , [FrameworkElement.UpdateDefaultStyle\(\)](#) ,  
[FrameworkElement.AddLogicalChild\(object\)](#) , [FrameworkElement.RemoveLogicalChild\(object\)](#) ,  
[FrameworkElement.Style](#) , [FrameworkElement.OverridesDefaultStyle](#) ,  
[FrameworkElement.UseLayoutRounding](#) , [FrameworkElement.DefaultStyleKey](#) ,  
[FrameworkElement.Triggers](#) , [FrameworkElement.TemplatedParent](#) , [FrameworkElement.Resources](#) ,  
[FrameworkElement.InheritanceBehavior](#) , [FrameworkElement.DataContext](#) ,  
[FrameworkElement.BindingGroup](#) , [FrameworkElement.Language](#) , [FrameworkElement.Name](#) ,  
[FrameworkElement.Tag](#) , [FrameworkElement.InputScope](#) , [FrameworkElement.ActualWidth](#) ,  
[FrameworkElement.ActualHeight](#) , [FrameworkElement.LayoutTransform](#) ,  
[FrameworkElement.Width](#) , [FrameworkElement.MinWidth](#) , [FrameworkElement.MaxWidth](#) ,  
[FrameworkElement.Height](#) , [FrameworkElement.MinHeight](#) , [FrameworkElement.MaxHeight](#) ,

[FrameworkElement.FlowDirection](#) , [FrameworkElement.Margin](#) ,  
[FrameworkElement.HorizontalAlignment](#) , [FrameworkElement.VerticalAlignment](#) ,  
[FrameworkElement.FocusVisualStyle](#) , [FrameworkElement.Cursor](#) , [FrameworkElement.ForceCursor](#) ,  
[FrameworkElement.IsInitialized](#) , [FrameworkElement.IsLoaded](#) , [FrameworkElement.ToolTip](#) ,  
[FrameworkElement.ContextMenu](#) , [FrameworkElement.Parent](#) , [FrameworkElement.TargetUpdated](#) ,  
[FrameworkElement.SourceUpdated](#) , [FrameworkElement.DataContextChanged](#) ,  
[FrameworkElement.RequestBringIntoView](#) , [FrameworkElement.SizeChanged](#) ,  
[FrameworkElement.Initialized](#) , [FrameworkElement.Loaded](#) , [FrameworkElement.Unloaded](#) ,  
[FrameworkElement.ToolTipOpening](#) , [FrameworkElement.ToolTipClosing](#) ,  
[FrameworkElement.ContextMenuOpening](#) , [FrameworkElement.ContextMenuClosing](#) ,  
[UIElement.PreviewMouseDownEvent](#) , [UIElement.MouseDownEvent](#) ,  
[UIElement.PreviewMouseUpEvent](#) , [UIElement.MouseUpEvent](#) ,  
[UIElement.PreviewMouseLeftButtonDownEvent](#) , [UIElement.MouseLeftButtonDownEvent](#) ,  
[UIElement.PreviewMouseLeftButtonUpEvent](#) , [UIElement.MouseLeftButtonUpEvent](#) ,  
[UIElement.PreviewMouseRightButtonDownEvent](#) , [UIElement.MouseRightButtonDownEvent](#) ,  
[UIElement.PreviewMouseRightButtonUpEvent](#) , [UIElement.MouseRightButtonUpEvent](#) ,  
[UIElement.PreviewMouseMoveEvent](#) , [UIElement.MouseMoveEvent](#) ,  
[UIElement.PreviewMouseWheelEvent](#) , [UIElement.MouseWheelEvent](#) , [UIElement.MouseEnterEvent](#) ,  
[UIElement.MouseLeaveEvent](#) , [UIElement.GotMouseCaptureEvent](#) ,  
[UIElement.LostMouseCaptureEvent](#) , [UIElement.QueryCursorEvent](#) ,  
[UIElement.PreviewStylusDownEvent](#) , [UIElement.StylusDownEvent](#) ,  
[UIElement.PreviewStylusUpEvent](#) , [UIElement.StylusUpEvent](#) , [UIElement.PreviewStylusMoveEvent](#) ,  
[UIElement.StylusMoveEvent](#) , [UIElement.PreviewStylusInAirMoveEvent](#) ,  
[UIElement.StylusInAirMoveEvent](#) , [UIElement.StylusEnterEvent](#) , [UIElement.StylusLeaveEvent](#) ,  
[UIElement.PreviewStylusInRangeEvent](#) , [UIElement.StylusInRangeEvent](#) ,  
[UIElement.PreviewStylusOutOfRangeEvent](#) , [UIElement.StylusOutOfRangeEvent](#) ,  
[UIElement.PreviewStylusSystemGestureEvent](#) , [UIElement.StylusSystemGestureEvent](#) ,  
[UIElement.GotStylusCaptureEvent](#) , [UIElement.LostStylusCaptureEvent](#) ,  
[UIElement.StylusButtonDownEvent](#) , [UIElement.StylusButtonUpEvent](#) ,  
[UIElement.PreviewStylusButtonDownEvent](#) , [UIElement.PreviewStylusButtonUpEvent](#) ,  
[UIElement.PreviewKeyDownEvent](#) , [UIElement.KeyDownEvent](#) , [UIElement.PreviewKeyUpEvent](#) ,  
[UIElement.KeyUpEvent](#) , [UIElement.PreviewGotKeyboardFocusEvent](#) ,  
[UIElement.GotKeyboardFocusEvent](#) , [UIElement.PreviewLostKeyboardFocusEvent](#) ,  
[UIElement.LostKeyboardFocusEvent](#) , [UIElement.PreviewTextInputEvent](#) , [UIElement.TextInputEvent](#) ,  
[UIElement.PreviewQueryContinueDragEvent](#) , [UIElement.QueryContinueDragEvent](#) ,  
[UIElement.PreviewGiveFeedbackEvent](#) , [UIElement.GiveFeedbackEvent](#) ,  
[UIElement.PreviewDragEnterEvent](#) , [UIElement.DragEnterEvent](#) , [UIElement.PreviewDragOverEvent](#) ,  
[UIElement.DragOverEvent](#) , [UIElement.PreviewDragLeaveEvent](#) , [UIElement.DragLeaveEvent](#) ,  
[UIElement.PreviewDropEvent](#) , [UIElement.DropEvent](#) , [UIElement.PreviewTouchDownEvent](#) ,  
[UIElement.TouchDownEvent](#) , [UIElement.PreviewTouchMoveEvent](#) , [UIElement.TouchMoveEvent](#) ,

[UIElement.PreviewTouchUpEvent](#) , [UIElement.TouchUpEvent](#) , [UIElement.GotTouchCaptureEvent](#) ,  
[UIElement.LostTouchCaptureEvent](#) , [UIElement.TouchEnterEvent](#) , [UIElement.TouchLeaveEvent](#) ,  
[UIElement.IsMouseDirectlyOverProperty](#) , [UIElement.IsMouseOverProperty](#) ,  
[UIElement.IsStylusOverProperty](#) , [UIElement.IsKeyboardFocusWithinProperty](#) ,  
[UIElement.IsMouseCapturedProperty](#) , [UIElement.IsMouseCaptureWithinProperty](#) ,  
[UIElement.IsStylusDirectlyOverProperty](#) , [UIElement.IsStylusCapturedProperty](#) ,  
[UIElement.IsStylusCaptureWithinProperty](#) , [UIElement.IsKeyboardFocusedProperty](#) ,  
[UIElement.AreAnyTouchesDirectlyOverProperty](#) , [UIElement.AreAnyTouchesOverProperty](#) ,  
[UIElement.AreAnyTouchesCapturedProperty](#) , [UIElement.AreAnyTouchesCapturedWithinProperty](#) ,  
[UIElement.AllowDropProperty](#) , [UIElement.RenderTransformProperty](#) ,  
[UIElement.RenderTransformOriginProperty](#) , [UIElement.OpacityProperty](#) ,  
[UIElement.OpacityMaskProperty](#) , [UIElement.BitmapEffectProperty](#) , [UIElement.EffectProperty](#) ,  
[UIElement.BitmapEffectInputProperty](#) , [UIElement.CacheModeProperty](#) , [UIElement.UidProperty](#) ,  
[UIElement.VisibilityProperty](#) , [UIElement.ClipToBoundsProperty](#) , [UIElement.ClipProperty](#) ,  
[UIElement.SnapsToDevicePixelsProperty](#) , [UIElement.GotFocusEvent](#) , [UIElement.LostFocusEvent](#) ,  
[UIElement.IsFocusedProperty](#) , [UIElement.IsEnabledProperty](#) , [UIElement.IsHitTestVisibleProperty](#) ,  
[UIElement.IsVisibleProperty](#) , [UIElement.FocusableProperty](#) ,  
[UIElement.IsManipulationEnabledProperty](#) , [UIElement.ManipulationStartingEvent](#) ,  
[UIElement.ManipulationStartedEvent](#) , [UIElement.ManipulationDeltaEvent](#) ,  
[UIElement.ManipulationInertiaStartingEvent](#) , [UIElement.ManipulationBoundaryFeedbackEvent](#) ,  
[UIElement.ManipulationCompletedEvent](#) ,  
[UIElement.ApplyAnimationClock\(DependencyProperty, AnimationClock\)](#) ,  
[UIElement.ApplyAnimationClock\(DependencyProperty, AnimationClock, HandoffBehavior\)](#) ,  
[UIElement.BeginAnimation\(DependencyProperty, AnimationTimeline\)](#) ,  
[UIElement.BeginAnimation\(DependencyProperty, AnimationTimeline, HandoffBehavior\)](#) ,  
[UIElement.GetAnimationBaseValue\(DependencyProperty\)](#) , [UIElement.RaiseEvent\(RoutedEventArgs\)](#) ,  
[UIElement.AddHandler\(RoutedEvent, Delegate\)](#) ,  
[UIElement.AddHandler\(RoutedEvent, Delegate, bool\)](#) ,  
[UIElement.RemoveHandler\(RoutedEvent, Delegate\)](#) ,  
[UIElement.AddToEventRoute\(EventRoute, RoutedEventArgs\)](#) ,  
[UIElement.OnPreviewMouseDown\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseDown\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseUp\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseUp\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseLeftButtonDown\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseLeftButtonDown\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseLeftButtonUp\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseLeftButtonUp\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseRightButtonDown\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseRightButtonDown\(MouseEventArgs\)](#) ,

[UIElement.OnPreviewMouseRightButtonUp\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseRightButtonUp\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseMove\(MouseEventArgs\)](#) , [UIElement.OnMouseMove\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseWheel\(MouseWheelEventArgs\)](#) ,  
[UIElement.OnMouseWheel\(MouseWheelEventArgs\)](#) , [UIElement.OnMouseEnter\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseLeave\(MouseEventArgs\)](#) , [UIElement.OnGotMouseCapture\(MouseEventArgs\)](#) ,  
[UIElement.OnLostMouseCapture\(MouseEventArgs\)](#) ,  
[UIElement.OnQueryCursor\(QueryCursorEventArgs\)](#) ,  
[UIElement.OnPreviewStylusDown\(StylusDownEventArgs\)](#) ,  
[UIElement.OnStylusDown\(StylusDownEventArgs\)](#) , [UIElement.OnPreviewStylusUp\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusUp\(StylusEventArgs\)](#) , [UIElement.OnPreviewStylusMove\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusMove\(StylusEventArgs\)](#) , [UIElement.OnPreviewStylusInAirMove\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusInAirMove\(StylusEventArgs\)](#) , [UIElement.OnStylusEnter\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusLeave\(StylusEventArgs\)](#) , [UIElement.OnPreviewStylusInRange\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusInRange\(StylusEventArgs\)](#) ,  
[UIElement.OnPreviewStylusOutOfRange\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusOutOfRange\(StylusEventArgs\)](#) ,  
[UIElement.OnPreviewStylusSystemGesture\(StylusSystemGestureEventArgs\)](#) ,  
[UIElement.OnStylusSystemGesture\(StylusSystemGestureEventArgs\)](#) ,  
[UIElement.OnGotStylusCapture\(StylusEventArgs\)](#) , [UIElement.OnLostStylusCapture\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusButtonDown\(StylusButtonEventArgs\)](#) ,  
[UIElement.OnStylusButtonUp\(StylusButtonEventArgs\)](#) ,  
[UIElement.OnPreviewStylusButtonDown\(StylusButtonEventArgs\)](#) ,  
[UIElement.OnPreviewStylusButtonUp\(StylusButtonEventArgs\)](#) ,  
[UIElement.OnPreviewKeyDown\(KeyEventArgs\)](#) , [UIElement.OnKeyDown\(KeyEventArgs\)](#) ,  
[UIElement.OnPreviewKeyUp\(KeyEventArgs\)](#) , [UIElement.OnKeyUp\(KeyEventArgs\)](#) ,  
[UIElement.OnPreviewGotKeyboardFocus\(KeyboardFocusChangedEventArgs\)](#) ,  
[UIElement.OnGotKeyboardFocus\(KeyboardFocusChangedEventArgs\)](#) ,  
[UIElement.OnPreviewLostKeyboardFocus\(KeyboardFocusChangedEventArgs\)](#) ,  
[UIElement.OnLostKeyboardFocus\(KeyboardFocusChangedEventArgs\)](#) ,  
[UIElement.OnPreviewTextInput\(TextCompositionEventArgs\)](#) ,  
[UIElement.OnTextInput\(TextCompositionEventArgs\)](#) ,  
[UIElement.OnPreviewQueryContinueDrag\(QueryContinueDragEventArgs\)](#) ,  
[UIElement.OnQueryContinueDrag\(QueryContinueDragEventArgs\)](#) ,  
[UIElement.OnPreviewGiveFeedback\(GiveFeedbackEventArgs\)](#) ,  
[UIElement.OnGiveFeedback\(GiveFeedbackEventArgs\)](#) ,  
[UIElement.OnPreviewDragEnter\(DragEventArgs\)](#) , [UIElement.OnDragEnter\(DragEventArgs\)](#) ,  
[UIElement.OnPreviewDragOver\(DragEventArgs\)](#) , [UIElement.OnDragOver\(DragEventArgs\)](#) ,  
[UIElement.OnPreviewDragLeave\(DragEventArgs\)](#) , [UIElement.OnDragLeave\(DragEventArgs\)](#) ,  
[UIElement.OnPreviewDrop\(DragEventArgs\)](#) , [UIElement.OnDrop\(DragEventArgs\)](#) ,

[UIElement.OnPreviewMouseDown\(TouchEventArgs\)](#) , [UIElement.OnMouseDown\(TouchEventArgs\)](#) ,  
[UIElement.OnPreviewMouseMove\(TouchEventArgs\)](#) , [UIElement.OnMouseMove\(TouchEventArgs\)](#) ,  
[UIElement.OnPreviewMouseUp\(TouchEventArgs\)](#) , [UIElement.OnMouseUp\(TouchEventArgs\)](#) ,  
[UIElement.OnGotTouchCapture\(TouchEventArgs\)](#) , [UIElement.OnLostTouchCapture\(TouchEventArgs\)](#) ,  
[UIElement.OnTouchEnter\(TouchEventArgs\)](#) , [UIElement.OnTouchLeave\(TouchEventArgs\)](#) ,  
[UIElement.OnIsMouseDirectlyOverChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsKeyboardFocusWithinChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsMouseCapturedChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsMouseCaptureWithinChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsStylusDirectlyOverChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsStylusCapturedChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsStylusCaptureWithinChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsKeyboardFocusedChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.InvalidateMeasure\(\)](#) , [UIElement.InvalidateArrange\(\)](#) , [UIElement.InvalidateVisual\(\)](#) ,  
[UIElement.OnChildDesiredSizeChanged\(UIElement\)](#) , [UIElement.Measure\(Size\)](#) ,  
[UIElement.Arrange\(Rect\)](#) , [UIElement.UpdateLayout\(\)](#) , [UIElement.TranslatePoint\(Point, UIElement\)](#) ,  
[UIElement.InputHitTest\(Point\)](#) , [UIElement.CaptureMouse\(\)](#) , [UIElement.ReleaseMouseCapture\(\)](#) ,  
[UIElement.CaptureStylus\(\)](#) , [UIElement.ReleaseStylusCapture\(\)](#) , [UIElement.Focus\(\)](#) ,  
[UIElement.OnAccessKey\(AccessKeyEventArgs\)](#) , [UIElement.HitTestCore\(PointHitTestParameters\)](#) ,  
[UIElement.HitTestCore\(GeometryHitTestParameters\)](#) , [UIElement.OnLostFocus\(RoutedEventArgs\)](#) ,  
[UIElement.OnCreateAutomationPeer\(\)](#) ,  
[UIElement.OnManipulationStarting\(ManipulationStartingEventArgs\)](#) ,  
[UIElement.OnManipulationStarted\(ManipulationStartedEventArgs\)](#) ,  
[UIElement.OnManipulationDelta\(ManipulationDeltaEventArgs\)](#) ,  
[UIElement.OnManipulationInertiaStarting\(ManipulationInertiaStartingEventArgs\)](#) ,  
[UIElement.OnManipulationBoundaryFeedback\(ManipulationBoundaryFeedbackEventArgs\)](#) ,  
[UIElement.OnManipulationCompleted\(ManipulationCompletedEventArgs\)](#) ,  
[UIElement.CaptureTouch\(TouchDevice\)](#) , [UIElement.ReleaseTouchCapture\(TouchDevice\)](#) ,  
[UIElement.ReleaseAllTouchCaptures\(\)](#) , [UIElement.HasAnimatedProperties](#) ,  
[UIElement.InputBindings](#) , [UIElement.CommandBindings](#) , [UIElement.AllowDrop](#) ,  
[UIElement.StylusPlugIns](#) , [UIElement.DesiredSize](#) , [UIElement.IsMeasureValid](#) ,  
[UIElement.IsArrangeValid](#) , [UIElement.RenderSize](#) , [UIElement.RenderTransform](#) ,  
[UIElement.RenderTransformOrigin](#) , [UIElement.IsMouseDirectlyOver](#) , [UIElement.IsMouseOver](#) ,  
[UIElement.IsStylusOver](#) , [UIElement.IsKeyboardFocusWithin](#) , [UIElement.IsMouseCaptured](#) ,  
[UIElement.IsMouseCaptureWithin](#) , [UIElement.IsStylusDirectlyOver](#) , [UIElement.IsStylusCaptured](#) ,  
[UIElement.IsStylusCaptureWithin](#) , [UIElement.IsKeyboardFocused](#) ,  
[UIElement.IsInputMethodEnabled](#) , [UIElement.Opacity](#) , [UIElement.OpacityMask](#) ,  
[UIElement.BitmapEffect](#) , [UIElement.Effect](#) , [UIElement.BitmapEffectInput](#) ,  
[UIElement.CacheMode](#) , [UIElement.Uid](#) , [UIElement.Visibility](#) , [UIElement.ClipToBounds](#) ,  
[UIElement.Clip](#) , [UIElement.SnapsToDevicePixels](#) , [UIElement.HasEffectiveKeyboardFocus](#) ,

[UIElement.IsFocused](#) , [UIElement.IsEnabled](#) , [UIElement.IsEnabledCore](#) ,  
[UIElement.IsHitTestVisible](#) , [UIElement.IsVisible](#) , [UIElement.Focusable](#) , [UIElement.PersistId](#) ,  
[UIElement.IsManipulationEnabled](#) , [UIElement.AreAnyTouchesOver](#) ,  
[UIElement.AreAnyTouchesDirectlyOver](#) , [UIElement.AreAnyTouchesCapturedWithin](#) ,  
[UIElement.AreAnyTouchesCaptured](#) , [UIElement.TouchesCaptured](#) ,  
[UIElement.TouchesCapturedWithin](#) , [UIElement.TouchesOver](#) , [UIElement.TouchesDirectlyOver](#) ,  
[UIElement.PreviewMouseDown](#) , [UIElement.MouseDown](#) , [UIElement.PreviewMouseUp](#) ,  
[UIElement.MouseUp](#) , [UIElement.PreviewMouseLeftButtonDown](#) ,  
[UIElement.MouseLeftButtonDown](#) , [UIElement.PreviewMouseLeftButtonUp](#) ,  
[UIElement.MouseLeftButtonUp](#) , [UIElement.PreviewMouseRightButtonDown](#) ,  
[UIElement.MouseRightButtonDown](#) , [UIElement.PreviewMouseRightButtonUp](#) ,  
[UIElement.MouseRightButtonUp](#) , [UIElement.PreviewMouseMove](#) , [UIElement.MouseMove](#) ,  
[UIElement.PreviewMouseWheel](#) , [UIElement.MouseWheel](#) , [UIElement.MouseEnter](#) ,  
[UIElement.MouseLeave](#) , [UIElement.GotMouseCapture](#) , [UIElement.LostMouseCapture](#) ,  
[UIElement.QueryCursor](#) , [UIElement.PreviewStylusDown](#) , [UIElement.StylusDown](#) ,  
[UIElement.PreviewStylusUp](#) , [UIElement.StylusUp](#) , [UIElement.PreviewStylusMove](#) ,  
[UIElement.StylusMove](#) , [UIElement.PreviewStylusInAirMove](#) , [UIElement.StylusInAirMove](#) ,  
[UIElement.StylusEnter](#) , [UIElement.StylusLeave](#) , [UIElement.PreviewStylusInRange](#) ,  
[UIElement.StylusInRange](#) , [UIElement.PreviewStylusOutOfRange](#) , [UIElement.StylusOutOfRange](#) ,  
[UIElement.PreviewStylusSystemGesture](#) , [UIElement.StylusSystemGesture](#) ,  
[UIElement.GotStylusCapture](#) , [UIElement.LostStylusCapture](#) , [UIElement.StylusButtonDown](#) ,  
[UIElement.StylusButtonUp](#) , [UIElement.PreviewStylusButtonDown](#) ,  
[UIElement.PreviewStylusButtonUp](#) , [UIElement.PreviewKeyDown](#) , [UIElement.KeyDown](#) ,  
[UIElement.PreviewKeyUp](#) , [UIElement.KeyUp](#) , [UIElement.PreviewGotKeyboardFocus](#) ,  
[UIElement.GotKeyboardFocus](#) , [UIElement.PreviewLostKeyboardFocus](#) ,  
[UIElement.LostKeyboardFocus](#) , [UIElement.PreviewTextInput](#) , [UIElement.TextInput](#) ,  
[UIElement.PreviewQueryContinueDrag](#) , [UIElement.QueryContinueDrag](#) ,  
[UIElement.PreviewGiveFeedback](#) , [UIElement.GiveFeedback](#) , [UIElement.PreviewDragEnter](#) ,  
[UIElement.DragEnter](#) , [UIElement.PreviewDragOver](#) , [UIElement.DragOver](#) ,  
[UIElement.PreviewDragLeave](#) , [UIElement.DragLeave](#) , [UIElement.PreviewDrop](#) , [UIElement.Drop](#) ,  
[UIElement.PreviewTouchDown](#) , [UIElement.TouchDown](#) , [UIElement.PreviewTouchMove](#) ,  
[UIElement.TouchMove](#) , [UIElement.PreviewTouchUp](#) , [UIElement.TouchUp](#) ,  
[UIElement.GotTouchCapture](#) , [UIElement.LostTouchCapture](#) , [UIElement.TouchEnter](#) ,  
[UIElement.TouchLeave](#) , [UIElement.IsMatchOverChanged](#) ,  
[UIElement.IsMatchKeyboardFocusWithinChanged](#) , [UIElement.IsMatchMouseCapturedChanged](#) ,  
[UIElement.IsMatchMouseCaptureWithinChanged](#) , [UIElement.IsMatchStylusDirectlyOverChanged](#) ,  
[UIElement.IsMatchStylusCapturedChanged](#) , [UIElement.IsMatchStylusCaptureWithinChanged](#) ,  
[UIElement.IsMatchKeyboardFocusedChanged](#) , [UIElement.LayoutUpdated](#) , [UIElement.GotFocus](#) ,  
[UIElement.LostFocus](#) , [UIElement.IsEnabledChanged](#) , [UIElement.IsMatchTestVisibleChanged](#) ,  
[UIElement.IsMatchVisibleChanged](#) , [UIElement.FocusableChanged](#) , [UIElement.ManipulationStarting](#) ,

[UIElement.ManipulationStarted](#) , [UIElement.ManipulationDelta](#) ,  
[UIElement.ManipulationInertiaStarting](#) , [UIElement.ManipulationBoundaryFeedback](#) ,  
[UIElement.ManipulationCompleted](#) , [Visual.AddVisualChild\(Visual\)](#) ,  
[Visual.RemoveVisualChild\(Visual\)](#) ,  
[Visual.OnVisualChildrenChanged\(DependencyObject, DependencyObject\)](#) ,  
[Visual.OnDpiChanged\(DpiScale, DpiScale\)](#) , [Visual.IsAncestorOf\(DependencyObject\)](#) ,  
[Visual.IsDescendantOf\(DependencyObject\)](#) , [Visual.FindCommonVisualAncestor\(DependencyObject\)](#) ,  
[Visual.TransformToAncestor\(Visual\)](#) , [Visual.TransformToAncestor\(Visual3D\)](#) ,  
[Visual.TransformToDescendant\(Visual\)](#) , [Visual.TransformToVisual\(Visual\)](#) ,  
[Visual.PointToScreen\(Point\)](#) , [Visual.PointFromScreen\(Point\)](#) , [Visual.VisualParent](#) ,  
[Visual.VisualTransform](#) , [Visual.VisualEffect](#) , [Visual.VisualBitmapEffect](#) ,  
[Visual.VisualBitmapEffectInput](#) , [Visual.VisualCacheMode](#) , [Visual.VisualScrollableAreaClip](#) ,  
[Visual.VisualClip](#) , [Visual.VisualOffset](#) , [Visual.VisualOpacity](#) , [Visual.VisualEdgeMode](#) ,  
[Visual.VisualBitmapScalingMode](#) , [Visual.VisualClearTypeHint](#) , [Visual.VisualTextRenderingMode](#) ,  
[Visual.VisualTextHintingMode](#) , [Visual.VisualOpacityMask](#) , [Visual.VisualXSnappingGuidelines](#) ,  
[Visual.VisualYSnappingGuidelines](#) , [DependencyObject.Equals\(object\)](#) ,  
[DependencyObject.GetHashCode\(\)](#) , [DependencyObject.GetValue\(DependencyProperty\)](#) ,  
[DependencyObject.SetValue\(DependencyProperty, object\)](#) ,  
[DependencyObject.SetCurrentValue\(DependencyProperty, object\)](#) ,  
[DependencyObject.SetValue\(DependencyPropertyKey, object\)](#) ,  
[DependencyObject.ClearValue\(DependencyProperty\)](#) ,  
[DependencyObject.ClearValue\(DependencyPropertyKey\)](#) ,  
[DependencyObject.CoerceValue\(DependencyProperty\)](#) ,  
[DependencyObject.InvalidateProperty\(DependencyProperty\)](#) ,  
[DependencyObject.ShouldSerializeProperty\(DependencyProperty\)](#) ,  
[DependencyObject.ReadLocalValue\(DependencyProperty\)](#) ,  
[DependencyObject.GetLocalValueEnumerator\(\)](#) , [DependencyObject.DependencyObjectType](#) ,  
[DependencyObject.IsSealed](#) , [DispatcherObject.Dispatcher](#) , [object.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class is typically used to inform the user that a long-running operation is in progress. The dialog is initialized with its default appearance and behavior.

## Constructors

[LoadingDialog\(\)](#)

Initializes a new instance of the [LoadingDialog](#) class.

```
public LoadingDialog()
```

## Remarks

This constructor sets up the loading dialog by initializing its components. Use this dialog to display a loading indicator during long-running operations.

## Methods

### InitializeComponent()

InitializeComponent

```
public void InitializeComponent()
```

# Class ProgressDialog

Namespace: [MBFWpfToolkit.Dialogs](#)

Assembly: MBFWpfToolkit.dll

Represents a dialog that displays progress information and allows updating its message dynamically.

```
public class ProgressDialog : Border, IAnimatable, IFrameworkInputElement, IInputElement,  
ISupportInitialize, IQueryAmbient, IAddChild, IUpdatableDialogContent, IComponentConnector
```

## Inheritance

```
object ↗ ← DispatcherObject ↗ ← DependencyObject ↗ ← Visual ↗ ← UIElement ↗ ←  
FrameworkElement ↗ ← Decorator ↗ ← Border ↗ ← ProgressDialog
```

## Implements

```
IAnimatable ↗ , IFrameworkInputElement ↗ , IInputElement ↗ , ISupportInitialize ↗ , IQueryAmbient ↗ ,  
IAddChild ↗ , IUpdatableDialogContent , IComponentConnector
```

## Inherited Members

```
Border.BorderThicknessProperty ↗ , Border.PaddingProperty ↗ , Border.CornerRadiusProperty ↗ ,  
Border.BorderBrushProperty ↗ , Border.BackgroundProperty ↗ , Border.MeasureOverride(Size) ↗ ,  
Border.ArrangeOverride(Size) ↗ , Border.OnRender(DrawingContext) ↗ , Border.BorderThickness ↗ ,  
Border.Padding ↗ , Border.CornerRadius ↗ , Border.BorderBrush ↗ , Border.Background ↗ ,  
Decorator.GetVisualChild(int) ↗ , Decorator.Child ↗ , Decorator.LogicalChildren ↗ ,  
Decorator.VisualChildrenCount ↗ , FrameworkElement.StyleProperty ↗ ,  
FrameworkElementOverridesDefaultStyleProperty ↗ , FrameworkElement.UseLayoutRoundingProperty ↗ ,  
FrameworkElement.DefaultStyleKeyProperty ↗ , FrameworkElement.DataContextProperty ↗ ,  
FrameworkElement.BindingGroupProperty ↗ , FrameworkElement.LanguageProperty ↗ ,  
FrameworkElement.NameProperty ↗ , FrameworkElement.TagProperty ↗ ,  
FrameworkElement.InputScopeProperty ↗ , FrameworkElement.RequestBringIntoViewEvent ↗ ,  
FrameworkElement.SizeChangedEvent ↗ , FrameworkElement.ActualWidthProperty ↗ ,  
FrameworkElement.ActualHeightProperty ↗ , FrameworkElement.LayoutTransformProperty ↗ ,  
FrameworkElement.WidthProperty ↗ , FrameworkElement.MinWidthProperty ↗ ,  
FrameworkElement.MaxWidthProperty ↗ , FrameworkElement.HeightProperty ↗ ,  
FrameworkElement.MinHeightProperty ↗ , FrameworkElement.MaxHeightProperty ↗ ,  
FrameworkElement.FlowDirectionProperty ↗ , FrameworkElement.MarginProperty ↗ ,  
FrameworkElement.HorizontalContentAlignmentProperty ↗ , FrameworkElement.VerticalAlignmentProperty ↗ ,  
FrameworkElement.FocusVisualStyleProperty ↗ , FrameworkElement.CursorProperty ↗ ,  
FrameworkElement.ForceCursorProperty ↗ , FrameworkElement.LoadedEvent ↗ ,  
FrameworkElement.UnloadedEvent ↗ , FrameworkElement.ToolTipProperty ↗ ,
```

[FrameworkElement.ContextMenuProperty](#) , [FrameworkElement.ToolTipOpeningEvent](#) ,  
[FrameworkElement.ToolTipClosingEvent](#) , [FrameworkElement.ContextMenuOpeningEvent](#) ,  
[FrameworkElement.ContextMenuClosingEvent](#) , [FrameworkElement.OnStyleChanged\(Style, Style\)](#) ,  
[FrameworkElement.ParentLayoutInvalidated\(UIElement\)](#) , [FrameworkElement.ApplyTemplate\(\)](#) ,  
[FrameworkElement.OnApplyTemplate\(\)](#) , [FrameworkElement.BeginStoryboard\(Storyboard\)](#) ,  
[FrameworkElement.BeginStoryboard\(Storyboard, HandoffBehavior\)](#) ,  
[FrameworkElement.BeginStoryboard\(Storyboard, HandoffBehavior, bool\)](#) ,  
[FrameworkElement.GetTemplateChild\(string\)](#) , [FrameworkElement.FindResource\(object\)](#) ,  
[FrameworkElement.TryFindResource\(object\)](#) ,  
[FrameworkElement.SetResourceReference\(DependencyProperty, object\)](#) ,  
[FrameworkElement.OnPropertyChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[FrameworkElement.OnVisualParentChanged\(DependencyObject\)](#) ,  
[FrameworkElement.GetBindingExpression\(DependencyProperty\)](#) ,  
[FrameworkElement.SetBinding\(DependencyProperty, BindingBase\)](#) ,  
[FrameworkElement.SetBinding\(DependencyProperty, string\)](#) , [FrameworkElement.GetUIParentCore\(\)](#) ,  
[FrameworkElement.BringIntoView\(\)](#) , [FrameworkElement.BringIntoView\(Rect\)](#) ,  
[FrameworkElement.GetFlowDirection\(DependencyObject\)](#) ,  
[FrameworkElement.SetFlowDirection\(DependencyObject, FlowDirection\)](#) ,  
[FrameworkElement.MeasureCore\(Size\)](#) , [FrameworkElement.ArrangeCore\(Rect\)](#) ,  
[FrameworkElement.OnRenderSizeChanged\(SizeChangedEventArgs\)](#) ,  
[FrameworkElement.GetLayoutClip\(Size\)](#) , [FrameworkElement.MoveFocus\(TraversalRequest\)](#) ,  
[FrameworkElement.PredictFocus\(FocusNavigationDirection\)](#) ,  
[FrameworkElement.OnGotFocus\(RoutedEventArgs\)](#) , [FrameworkElement.BeginInit\(\)](#) ,  
[FrameworkElement.EndInit\(\)](#) , [FrameworkElement.OnInitialized\(EventArgs\)](#) ,  
[FrameworkElement.OnToolTipOpening\(ToolTipEventArgs\)](#) ,  
[FrameworkElement.OnToolTipClosing\(ToolTipEventArgs\)](#) ,  
[FrameworkElement.OnContextMenuOpening\(ContextMenuEventArgs\)](#) ,  
[FrameworkElement.OnContextMenuClosing\(ContextMenuEventArgs\)](#) ,  
[FrameworkElement.RegisterName\(string, object\)](#) , [FrameworkElement.UnregisterName\(string\)](#) ,  
[FrameworkElement.FindName\(string\)](#) , [FrameworkElement.UpdateDefaultStyle\(\)](#) ,  
[FrameworkElement.AddLogicalChild\(object\)](#) , [FrameworkElement.RemoveLogicalChild\(object\)](#) ,  
[FrameworkElement.Style](#) , [FrameworkElement.OverridesDefaultStyle](#) ,  
[FrameworkElement.UseLayoutRounding](#) , [FrameworkElement.DefaultStyleKey](#) ,  
[FrameworkElement.Triggers](#) , [FrameworkElement.TemplatedParent](#) , [FrameworkElement.Resources](#) ,  
[FrameworkElement.InheritanceBehavior](#) , [FrameworkElement.DataContext](#) ,  
[FrameworkElement.BindingGroup](#) , [FrameworkElement.Language](#) , [FrameworkElement.Name](#) ,  
[FrameworkElement.Tag](#) , [FrameworkElement.InputScope](#) , [FrameworkElement.ActualWidth](#) ,  
[FrameworkElement.ActualHeight](#) , [FrameworkElement.LayoutTransform](#) ,  
[FrameworkElement.Width](#) , [FrameworkElement.MinWidth](#) , [FrameworkElement.MaxWidth](#) ,  
[FrameworkElement.Height](#) , [FrameworkElement.MinHeight](#) , [FrameworkElement.MaxHeight](#) ,

[FrameworkElement.FlowDirection](#) , [FrameworkElement.Margin](#) ,  
[FrameworkElement.HorizontalAlignment](#) , [FrameworkElement.VerticalAlignment](#) ,  
[FrameworkElement.FocusVisualStyle](#) , [FrameworkElement.Cursor](#) , [FrameworkElement.ForceCursor](#) ,  
[FrameworkElement.IsInitialized](#) , [FrameworkElement.IsLoaded](#) , [FrameworkElement.ToolTip](#) ,  
[FrameworkElement.ContextMenu](#) , [FrameworkElement.Parent](#) , [FrameworkElement.TargetUpdated](#) ,  
[FrameworkElement.SourceUpdated](#) , [FrameworkElement.DataContextChanged](#) ,  
[FrameworkElement.RequestBringIntoView](#) , [FrameworkElement.SizeChanged](#) ,  
[FrameworkElement.Initialized](#) , [FrameworkElement.Loaded](#) , [FrameworkElement.Unloaded](#) ,  
[FrameworkElement.ToolTipOpening](#) , [FrameworkElement.ToolTipClosing](#) ,  
[FrameworkElement.ContextMenuOpening](#) , [FrameworkElement.ContextMenuClosing](#) ,  
[UIElement.PreviewMouseDownEvent](#) , [UIElement.MouseDownEvent](#) ,  
[UIElement.PreviewMouseUpEvent](#) , [UIElement.MouseUpEvent](#) ,  
[UIElement.PreviewMouseLeftButtonDownEvent](#) , [UIElement.MouseLeftButtonDownEvent](#) ,  
[UIElement.PreviewMouseLeftButtonUpEvent](#) , [UIElement.MouseLeftButtonUpEvent](#) ,  
[UIElement.PreviewMouseRightButtonDownEvent](#) , [UIElement.MouseRightButtonDownEvent](#) ,  
[UIElement.PreviewMouseRightButtonUpEvent](#) , [UIElement.MouseRightButtonUpEvent](#) ,  
[UIElement.PreviewMouseMoveEvent](#) , [UIElement.MouseMoveEvent](#) ,  
[UIElement.PreviewMouseWheelEvent](#) , [UIElement.MouseWheelEvent](#) , [UIElement.MouseEnterEvent](#) ,  
[UIElement.MouseLeaveEvent](#) , [UIElement.GotMouseCaptureEvent](#) ,  
[UIElement.LostMouseCaptureEvent](#) , [UIElement.QueryCursorEvent](#) ,  
[UIElement.PreviewStylusDownEvent](#) , [UIElement.StylusDownEvent](#) ,  
[UIElement.PreviewStylusUpEvent](#) , [UIElement.StylusUpEvent](#) , [UIElement.PreviewStylusMoveEvent](#) ,  
[UIElement.StylusMoveEvent](#) , [UIElement.PreviewStylusInAirMoveEvent](#) ,  
[UIElement.StylusInAirMoveEvent](#) , [UIElement.StylusEnterEvent](#) , [UIElement.StylusLeaveEvent](#) ,  
[UIElement.PreviewStylusInRangeEvent](#) , [UIElement.StylusInRangeEvent](#) ,  
[UIElement.PreviewStylusOutOfRangeEvent](#) , [UIElement.StylusOutOfRangeEvent](#) ,  
[UIElement.PreviewStylusSystemGestureEvent](#) , [UIElement.StylusSystemGestureEvent](#) ,  
[UIElement.GotStylusCaptureEvent](#) , [UIElement.LostStylusCaptureEvent](#) ,  
[UIElement.StylusButtonDownEvent](#) , [UIElement.StylusButtonUpEvent](#) ,  
[UIElement.PreviewStylusButtonDownEvent](#) , [UIElement.PreviewStylusButtonUpEvent](#) ,  
[UIElement.PreviewKeyDownEvent](#) , [UIElement.KeyDownEvent](#) , [UIElement.PreviewKeyUpEvent](#) ,  
[UIElement.KeyUpEvent](#) , [UIElement.PreviewGotKeyboardFocusEvent](#) ,  
[UIElement.GotKeyboardFocusEvent](#) , [UIElement.PreviewLostKeyboardFocusEvent](#) ,  
[UIElement.LostKeyboardFocusEvent](#) , [UIElement.PreviewTextInputEvent](#) , [UIElement.TextInputEvent](#) ,  
[UIElement.PreviewQueryContinueDragEvent](#) , [UIElement.QueryContinueDragEvent](#) ,  
[UIElement.PreviewGiveFeedbackEvent](#) , [UIElement.GiveFeedbackEvent](#) ,  
[UIElement.PreviewDragEnterEvent](#) , [UIElement.DragEnterEvent](#) , [UIElement.PreviewDragOverEvent](#) ,  
[UIElement.DragOverEvent](#) , [UIElement.PreviewDragLeaveEvent](#) , [UIElement.DragLeaveEvent](#) ,  
[UIElement.PreviewDropEvent](#) , [UIElement.DropEvent](#) , [UIElement.PreviewTouchDownEvent](#) ,  
[UIElement.TouchDownEvent](#) , [UIElement.PreviewTouchMoveEvent](#) , [UIElement.TouchMoveEvent](#) ,

[UIElement.PreviewTouchUpEvent](#) , [UIElement.TouchUpEvent](#) , [UIElement.GotTouchCaptureEvent](#) ,  
[UIElement.LostTouchCaptureEvent](#) , [UIElement.TouchEnterEvent](#) , [UIElement.TouchLeaveEvent](#) ,  
[UIElement.IsMouseDirectlyOverProperty](#) , [UIElement.IsMouseOverProperty](#) ,  
[UIElement.IsStylusOverProperty](#) , [UIElement.IsKeyboardFocusWithinProperty](#) ,  
[UIElement.IsMouseCapturedProperty](#) , [UIElement.IsMouseCaptureWithinProperty](#) ,  
[UIElement.IsStylusDirectlyOverProperty](#) , [UIElement.IsStylusCapturedProperty](#) ,  
[UIElement.IsStylusCaptureWithinProperty](#) , [UIElement.IsKeyboardFocusedProperty](#) ,  
[UIElement.AreAnyTouchesDirectlyOverProperty](#) , [UIElement.AreAnyTouchesOverProperty](#) ,  
[UIElement.AreAnyTouchesCapturedProperty](#) , [UIElement.AreAnyTouchesCapturedWithinProperty](#) ,  
[UIElement.AllowDropProperty](#) , [UIElement.RenderTransformProperty](#) ,  
[UIElement.RenderTransformOriginProperty](#) , [UIElement.OpacityProperty](#) ,  
[UIElement.OpacityMaskProperty](#) , [UIElement.BitmapEffectProperty](#) , [UIElement.EffectProperty](#) ,  
[UIElement.BitmapEffectInputProperty](#) , [UIElement.CacheModeProperty](#) , [UIElement.UidProperty](#) ,  
[UIElement.VisibilityProperty](#) , [UIElement.ClipToBoundsProperty](#) , [UIElement.ClipProperty](#) ,  
[UIElement.SnapsToDevicePixelsProperty](#) , [UIElement.GotFocusEvent](#) , [UIElement.LostFocusEvent](#) ,  
[UIElement.FocusedProperty](#) , [UIElement.IsEnabledProperty](#) , [UIElement.HitTestVisibleProperty](#) ,  
[UIElement.VisibleProperty](#) , [UIElement.FocusableProperty](#) ,  
[UIElement.ManipulationEnabledProperty](#) , [UIElement.ManipulationStartingEvent](#) ,  
[UIElement.ManipulationStartedEvent](#) , [UIElement.ManipulationDeltaEvent](#) ,  
[UIElement.ManipulationInertiaStartingEvent](#) , [UIElement.ManipulationBoundaryFeedbackEvent](#) ,  
[UIElement.ManipulationCompletedEvent](#) ,  
[UIElement.ApplyAnimationClock\(DependencyProperty, AnimationClock\)](#) ,  
[UIElement.ApplyAnimationClock\(DependencyProperty, AnimationClock, HandoffBehavior\)](#) ,  
[UIElement.BeginAnimation\(DependencyProperty, AnimationTimeline\)](#) ,  
[UIElement.BeginAnimation\(DependencyProperty, AnimationTimeline, HandoffBehavior\)](#) ,  
[UIElement.GetAnimationBaseValue\(DependencyProperty\)](#) , [UIElement.RaiseEvent\(RoutedEventArgs\)](#) ,  
[UIElement.AddHandler\(RoutedEvent, Delegate\)](#) ,  
[UIElement.AddHandler\(RoutedEvent, Delegate, bool\)](#) ,  
[UIElement.RemoveHandler\(RoutedEvent, Delegate\)](#) ,  
[UIElement.AddToEventRoute\(EventRoute, RoutedEventArgs\)](#) ,  
[UIElement.OnPreviewMouseDown\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseDown\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseUp\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseUp\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseLeftButtonDown\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseLeftButtonDown\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseLeftButtonUp\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseLeftButtonUp\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseRightButtonDown\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseRightButtonDown\(MouseEventArgs\)](#) ,

[UIElement.OnPreviewMouseRightButtonUp\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseRightButtonUp\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseMove\(MouseEventArgs\)](#) , [UIElement.OnMouseMove\(MouseEventArgs\)](#) ,  
[UIElement.OnPreviewMouseWheel\(MouseWheelEventArgs\)](#) ,  
[UIElement.OnMouseWheel\(MouseWheelEventArgs\)](#) , [UIElement.OnMouseEnter\(MouseEventArgs\)](#) ,  
[UIElement.OnMouseLeave\(MouseEventArgs\)](#) , [UIElement.OnGotMouseCapture\(MouseEventArgs\)](#) ,  
[UIElement.OnLostMouseCapture\(MouseEventArgs\)](#) ,  
[UIElement.OnQueryCursor\(QueryCursorEventArgs\)](#) ,  
[UIElement.OnPreviewStylusDown\(StylusDownEventArgs\)](#) ,  
[UIElement.OnStylusDown\(StylusDownEventArgs\)](#) , [UIElement.OnPreviewStylusUp\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusUp\(StylusEventArgs\)](#) , [UIElement.OnPreviewStylusMove\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusMove\(StylusEventArgs\)](#) , [UIElement.OnPreviewStylusInAirMove\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusInAirMove\(StylusEventArgs\)](#) , [UIElement.OnStylusEnter\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusLeave\(StylusEventArgs\)](#) , [UIElement.OnPreviewStylusInRange\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusInRange\(StylusEventArgs\)](#) ,  
[UIElement.OnPreviewStylusOutOfRange\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusOutOfRange\(StylusEventArgs\)](#) ,  
[UIElement.OnPreviewStylusSystemGesture\(StylusSystemGestureEventArgs\)](#) ,  
[UIElement.OnStylusSystemGesture\(StylusSystemGestureEventArgs\)](#) ,  
[UIElement.OnGotStylusCapture\(StylusEventArgs\)](#) , [UIElement.OnLostStylusCapture\(StylusEventArgs\)](#) ,  
[UIElement.OnStylusButtonDown\(StylusButtonEventArgs\)](#) ,  
[UIElement.OnStylusButtonUp\(StylusButtonEventArgs\)](#) ,  
[UIElement.OnPreviewStylusButtonDown\(StylusButtonEventArgs\)](#) ,  
[UIElement.OnPreviewStylusButtonUp\(StylusButtonEventArgs\)](#) ,  
[UIElement.OnPreviewKeyDown\(KeyEventArgs\)](#) , [UIElement.OnKeyDown\(KeyEventArgs\)](#) ,  
[UIElement.OnPreviewKeyUp\(KeyEventArgs\)](#) , [UIElement.OnKeyUp\(KeyEventArgs\)](#) ,  
[UIElement.OnPreviewGotKeyboardFocus\(KeyboardFocusChangedEventArgs\)](#) ,  
[UIElement.OnGotKeyboardFocus\(KeyboardFocusChangedEventArgs\)](#) ,  
[UIElement.OnPreviewLostKeyboardFocus\(KeyboardFocusChangedEventArgs\)](#) ,  
[UIElement.OnLostKeyboardFocus\(KeyboardFocusChangedEventArgs\)](#) ,  
[UIElement.OnPreviewTextInput\(TextCompositionEventArgs\)](#) ,  
[UIElement.OnTextInput\(TextCompositionEventArgs\)](#) ,  
[UIElement.OnPreviewQueryContinueDrag\(QueryContinueDragEventArgs\)](#) ,  
[UIElement.OnQueryContinueDrag\(QueryContinueDragEventArgs\)](#) ,  
[UIElement.OnPreviewGiveFeedback\(GiveFeedbackEventArgs\)](#) ,  
[UIElement.OnGiveFeedback\(GiveFeedbackEventArgs\)](#) ,  
[UIElement.OnPreviewDragEnter\(DragEventArgs\)](#) , [UIElement.OnDragEnter\(DragEventArgs\)](#) ,  
[UIElement.OnPreviewDragOver\(DragEventArgs\)](#) , [UIElement.OnDragOver\(DragEventArgs\)](#) ,  
[UIElement.OnPreviewDragLeave\(DragEventArgs\)](#) , [UIElement.OnDragLeave\(DragEventArgs\)](#) ,  
[UIElement.OnPreviewDrop\(DragEventArgs\)](#) , [UIElement.OnDrop\(DragEventArgs\)](#) ,

[UIElement.OnPreviewMouseDown\(TouchEventArgs\)](#) , [UIElement.OnMouseDown\(TouchEventArgs\)](#) ,  
[UIElement.OnPreviewMouseMove\(TouchEventArgs\)](#) , [UIElement.OnMouseMove\(TouchEventArgs\)](#) ,  
[UIElement.OnPreviewMouseUp\(TouchEventArgs\)](#) , [UIElement.OnMouseUp\(TouchEventArgs\)](#) ,  
[UIElement.OnGotTouchCapture\(TouchEventArgs\)](#) , [UIElement.OnLostTouchCapture\(TouchEventArgs\)](#) ,  
[UIElement.OnTouchEnter\(TouchEventArgs\)](#) , [UIElement.OnTouchLeave\(TouchEventArgs\)](#) ,  
[UIElement.OnIsMouseDirectlyOverChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsKeyboardFocusWithinChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsMouseCapturedChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsMouseCaptureWithinChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsStylusDirectlyOverChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsStylusCapturedChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsStylusCaptureWithinChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.OnIsKeyboardFocusedChanged\(DependencyPropertyChangedEventArgs\)](#) ,  
[UIElement.InvalidateMeasure\(\)](#) , [UIElement.InvalidateArrange\(\)](#) , [UIElement.InvalidateVisual\(\)](#) ,  
[UIElement.OnChildDesiredSizeChanged\(UIElement\)](#) , [UIElement.Measure\(Size\)](#) ,  
[UIElement.Arrange\(Rect\)](#) , [UIElement.UpdateLayout\(\)](#) , [UIElement.TranslatePoint\(Point, UIElement\)](#) ,  
[UIElement.InputHitTest\(Point\)](#) , [UIElement.CaptureMouse\(\)](#) , [UIElement.ReleaseMouseCapture\(\)](#) ,  
[UIElement.CaptureStylus\(\)](#) , [UIElement.ReleaseStylusCapture\(\)](#) , [UIElement.Focus\(\)](#) ,  
[UIElement.OnAccessKey\(AccessKeyEventArgs\)](#) , [UIElement.HitTestCore\(PointHitTestParameters\)](#) ,  
[UIElement.HitTestCore\(GeometryHitTestParameters\)](#) , [UIElement.OnLostFocus\(RoutedEventArgs\)](#) ,  
[UIElement.OnCreateAutomationPeer\(\)](#) ,  
[UIElement.OnManipulationStarting\(ManipulationStartingEventArgs\)](#) ,  
[UIElement.OnManipulationStarted\(ManipulationStartedEventArgs\)](#) ,  
[UIElement.OnManipulationDelta\(ManipulationDeltaEventArgs\)](#) ,  
[UIElement.OnManipulationInertiaStarting\(ManipulationInertiaStartingEventArgs\)](#) ,  
[UIElement.OnManipulationBoundaryFeedback\(ManipulationBoundaryFeedbackEventArgs\)](#) ,  
[UIElement.OnManipulationCompleted\(ManipulationCompletedEventArgs\)](#) ,  
[UIElement.CaptureTouch\(TouchDevice\)](#) , [UIElement.ReleaseTouchCapture\(TouchDevice\)](#) ,  
[UIElement.ReleaseAllTouchCaptures\(\)](#) , [UIElement.HasAnimatedProperties](#) ,  
[UIElement.InputBindings](#) , [UIElement.CommandBindings](#) , [UIElement.AllowDrop](#) ,  
[UIElement.StylusPlugIns](#) , [UIElement.DesiredSize](#) , [UIElement.IsMeasureValid](#) ,  
[UIElement.IsArrangeValid](#) , [UIElement.RenderSize](#) , [UIElement.RenderTransform](#) ,  
[UIElement.RenderTransformOrigin](#) , [UIElement.IsMouseDirectlyOver](#) , [UIElement.IsMouseOver](#) ,  
[UIElement.IsStylusOver](#) , [UIElement.IsKeyboardFocusWithin](#) , [UIElement.IsMouseCaptured](#) ,  
[UIElement.IsMouseCaptureWithin](#) , [UIElement.IsStylusDirectlyOver](#) , [UIElement.IsStylusCaptured](#) ,  
[UIElement.IsStylusCaptureWithin](#) , [UIElement.IsKeyboardFocused](#) ,  
[UIElement.IsInputMethodEnabled](#) , [UIElement.Opacity](#) , [UIElement.OpacityMask](#) ,  
[UIElement.BitmapEffect](#) , [UIElement.Effect](#) , [UIElement.BitmapEffectInput](#) ,  
[UIElement.CacheMode](#) , [UIElement.Uid](#) , [UIElement.Visibility](#) , [UIElement.ClipToBounds](#) ,  
[UIElement.Clip](#) , [UIElement.SnapsToDevicePixels](#) , [UIElement.HasEffectiveKeyboardFocus](#) ,

[UIElement.IsFocused](#) , [UIElement.IsEnabled](#) , [UIElement.IsEnabledCore](#) ,  
[UIElement.IsHitTestVisible](#) , [UIElement.IsVisible](#) , [UIElement.Focusable](#) , [UIElement.PersistId](#) ,  
[UIElement.IsManipulationEnabled](#) , [UIElement.AreAnyTouchesOver](#) ,  
[UIElement.AreAnyTouchesDirectlyOver](#) , [UIElement.AreAnyTouchesCapturedWithin](#) ,  
[UIElement.AreAnyTouchesCaptured](#) , [UIElement.TouchesCaptured](#) ,  
[UIElement.TouchesCapturedWithin](#) , [UIElement.TouchesOver](#) , [UIElement.TouchesDirectlyOver](#) ,  
[UIElement.PreviewMouseDown](#) , [UIElement.MouseDown](#) , [UIElement.PreviewMouseUp](#) ,  
[UIElement.MouseUp](#) , [UIElement.PreviewMouseLeftButtonDown](#) ,  
[UIElement.MouseLeftButtonDown](#) , [UIElement.PreviewMouseLeftButtonUp](#) ,  
[UIElement.MouseLeftButtonUp](#) , [UIElement.PreviewMouseRightButtonDown](#) ,  
[UIElement.MouseRightButtonDown](#) , [UIElement.PreviewMouseRightButtonUp](#) ,  
[UIElement.MouseRightButtonUp](#) , [UIElement.PreviewMouseMove](#) , [UIElement.MouseMove](#) ,  
[UIElement.PreviewMouseWheel](#) , [UIElement.MouseWheel](#) , [UIElement.MouseEnter](#) ,  
[UIElement.MouseLeave](#) , [UIElement.GotMouseCapture](#) , [UIElement.LostMouseCapture](#) ,  
[UIElement.QueryCursor](#) , [UIElement.PreviewStylusDown](#) , [UIElement.StylusDown](#) ,  
[UIElement.PreviewStylusUp](#) , [UIElement.StylusUp](#) , [UIElement.PreviewStylusMove](#) ,  
[UIElement.StylusMove](#) , [UIElement.PreviewStylusInAirMove](#) , [UIElement.StylusInAirMove](#) ,  
[UIElement.StylusEnter](#) , [UIElement.StylusLeave](#) , [UIElement.PreviewStylusInRange](#) ,  
[UIElement.StylusInRange](#) , [UIElement.PreviewStylusOutOfRange](#) , [UIElement.StylusOutOfRange](#) ,  
[UIElement.PreviewStylusSystemGesture](#) , [UIElement.StylusSystemGesture](#) ,  
[UIElement.GotStylusCapture](#) , [UIElement.LostStylusCapture](#) , [UIElement.StylusButtonDown](#) ,  
[UIElement.StylusButtonUp](#) , [UIElement.PreviewStylusButtonDown](#) ,  
[UIElement.PreviewStylusButtonUp](#) , [UIElement.PreviewKeyDown](#) , [UIElement.KeyDown](#) ,  
[UIElement.PreviewKeyUp](#) , [UIElement.KeyUp](#) , [UIElement.PreviewGotKeyboardFocus](#) ,  
[UIElement.GotKeyboardFocus](#) , [UIElement.PreviewLostKeyboardFocus](#) ,  
[UIElement.LostKeyboardFocus](#) , [UIElement.PreviewTextInput](#) , [UIElement.TextInput](#) ,  
[UIElement.PreviewQueryContinueDrag](#) , [UIElement.QueryContinueDrag](#) ,  
[UIElement.PreviewGiveFeedback](#) , [UIElement.GiveFeedback](#) , [UIElement.PreviewDragEnter](#) ,  
[UIElement.DragEnter](#) , [UIElement.PreviewDragOver](#) , [UIElement.DragOver](#) ,  
[UIElement.PreviewDragLeave](#) , [UIElement.DragLeave](#) , [UIElement.PreviewDrop](#) , [UIElement.Drop](#) ,  
[UIElement.PreviewTouchDown](#) , [UIElement.TouchDown](#) , [UIElement.PreviewTouchMove](#) ,  
[UIElement.TouchMove](#) , [UIElement.PreviewTouchUp](#) , [UIElement.TouchUp](#) ,  
[UIElement.GotTouchCapture](#) , [UIElement.LostTouchCapture](#) , [UIElement.TouchEnter](#) ,  
[UIElement.TouchLeave](#) , [UIElement.IsMatchOverChanged](#) ,  
[UIElement.IsMatchKeyboardFocusWithinChanged](#) , [UIElement.IsMatchMouseCapturedChanged](#) ,  
[UIElement.IsMatchMouseCaptureWithinChanged](#) , [UIElement.IsMatchStylusDirectlyOverChanged](#) ,  
[UIElement.IsMatchStylusCapturedChanged](#) , [UIElement.IsMatchStylusCaptureWithinChanged](#) ,  
[UIElement.IsMatchKeyboardFocusedChanged](#) , [UIElement.LayoutUpdated](#) , [UIElement.GotFocus](#) ,  
[UIElement.LostFocus](#) , [UIElement.IsEnabledChanged](#) , [UIElement.IsMatchTestVisibleChanged](#) ,  
[UIElement.IsMatchVisibleChanged](#) , [UIElement.FocusableChanged](#) , [UIElement.ManipulationStarting](#) ,

[UIElement.ManipulationStarted](#) , [UIElement.ManipulationDelta](#) ,  
[UIElement.ManipulationInertiaStarting](#) , [UIElement.ManipulationBoundaryFeedback](#) ,  
[UIElement.ManipulationCompleted](#) , [Visual.AddVisualChild\(Visual\)](#) ,  
[Visual.RemoveVisualChild\(Visual\)](#) ,  
[Visual.OnVisualChildrenChanged\(DependencyObject, DependencyObject\)](#) ,  
[Visual.OnDpiChanged\(DpiScale, DpiScale\)](#) , [Visual.IsAncestorOf\(DependencyObject\)](#) ,  
[Visual.IsDescendantOf\(DependencyObject\)](#) , [Visual.FindCommonVisualAncestor\(DependencyObject\)](#) ,  
[Visual.TransformToAncestor\(Visual\)](#) , [Visual.TransformToAncestor\(Visual3D\)](#) ,  
[Visual.TransformToDescendant\(Visual\)](#) , [Visual.TransformToVisual\(Visual\)](#) ,  
[Visual.PointToScreen\(Point\)](#) , [Visual.PointFromScreen\(Point\)](#) , [Visual.VisualParent](#) ,  
[Visual.VisualTransform](#) , [Visual.VisualEffect](#) , [Visual.VisualBitmapEffect](#) ,  
[Visual.VisualBitmapEffectInput](#) , [Visual.VisualCacheMode](#) , [Visual.VisualScrollableAreaClip](#) ,  
[Visual.VisualClip](#) , [Visual.VisualOffset](#) , [Visual.VisualOpacity](#) , [Visual.VisualEdgeMode](#) ,  
[Visual.VisualBitmapScalingMode](#) , [Visual.VisualClearTypeHint](#) , [Visual.VisualTextRenderingMode](#) ,  
[Visual.VisualTextHintingMode](#) , [Visual.VisualOpacityMask](#) , [Visual.VisualXSnappingGuidelines](#) ,  
[Visual.VisualYSnappingGuidelines](#) , [DependencyObject.Equals\(object\)](#) ,  
[DependencyObject.GetHashCode\(\)](#) , [DependencyObject.GetValue\(DependencyProperty\)](#) ,  
[DependencyObject.SetValue\(DependencyProperty, object\)](#) ,  
[DependencyObject.SetCurrentValue\(DependencyProperty, object\)](#) ,  
[DependencyObject.SetValue\(DependencyPropertyKey, object\)](#) ,  
[DependencyObject.ClearValue\(DependencyProperty\)](#) ,  
[DependencyObject.ClearValue\(DependencyPropertyKey\)](#) ,  
[DependencyObject.CoerceValue\(DependencyProperty\)](#) ,  
[DependencyObject.InvalidateProperty\(DependencyProperty\)](#) ,  
[DependencyObject.ShouldSerializeProperty\(DependencyProperty\)](#) ,  
[DependencyObject.ReadLocalValue\(DependencyProperty\)](#) ,  
[DependencyObject.GetLocalValueEnumerator\(\)](#) , [DependencyObject.DependencyObjectType](#) ,  
[DependencyObject.IsSealed](#) , [DispatcherObject.Dispatcher](#) , [object.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This class provides functionality for displaying a progress dialog with a description message. The message can be updated at runtime using the [UpdateMessage\(string\)](#) method.

## Constructors

### ProgressDialog()

Initializes a new instance of the [ProgressDialog](#) class.

```
public ProgressDialog()
```

## Remarks

This constructor sets up the progress dialog by initializing its components. Use this dialog to display progress information during long-running operations.

# Methods

## InitializeComponent()

InitializeComponent

```
public void InitializeComponent()
```

## UpdateMessage(string)

Updates the text displayed in the description control.

```
public void UpdateMessage(string message)
```

### Parameters

**message** [string](#)

The new message to display. Cannot be null or empty.

## Remarks

This method sets the text of the description control to the specified message. Ensure that **message** contains meaningful content to display.

# Namespace MBFWpfToolkit.Extensions

## Classes

### [CollectionExtensions](#)

Provides extension methods for working with collections.

### [ListExtensions](#)

Provides extension methods for [List<T>](#) to support additional operations.

### [ToleranceGroup< TKey, TElement >](#)

Represents a collection of elements that share a common key, with additional functionality for grouping and list operations.

# Class CollectionExtensions

Namespace: [MBFWpfToolkit.Extensions](#)

Assembly: MBFWpfToolkit.dll

Provides extension methods for working with collections.

```
public static class CollectionExtensions
```

## Inheritance

[object](#) ← CollectionExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### GroupByWithTolerance<T>(IEnumerable<T>, Func<T, double>, double)

Groups elements of a sequence based on a numeric key and a specified tolerance.

```
public static IEnumerable<IGrouping<double, T>> GroupByWithTolerance<T>(this IEnumerable<T>  
source, Func<T, double> keySelector, double tolerance)
```

#### Parameters

**source** [IEnumerable](#)<T>

The sequence of elements to group. Cannot be null.

**keySelector** [Func](#)<T, [double](#)>

A function to extract the numeric key from each element. Cannot be null.

**tolerance** [double](#)

The maximum allowable difference between keys for elements to be grouped together. Must be a non-negative value.

## Returns

[IEnumerable](#) <[IGrouping](#) <[double](#), T>>

A sequence of groupings, where each grouping contains elements whose keys differ by no more than the specified tolerance.

## Type Parameters

T

The type of the elements in the source sequence.

## Remarks

This method first sorts the elements by their numeric keys and then groups them based on the specified tolerance. The keys are rounded to three decimal places for comparison.

# ToObservableCollection<T>(IEnumerable<T>)

Converts an enumerable collection to an ObservableCollection.

```
public static ObservableCollection<T> ToObservableCollection<T>(this IEnumerable<T> source)
```

## Parameters

source [IEnumerable](#)<T>

The source enumerable collection to convert.

## Returns

[ObservableCollection](#) <T>

An ObservableCollection containing the elements from the source.

## Type Parameters

T

The type of elements in the collection.

# Class ListExtensions

Namespace: [MBFWpfToolkit.Extensions](#)

Assembly: MBFWpfToolkit.dll

Provides extension methods for [List<T>](#) to support additional operations.

```
public static class ListExtensions
```

## Inheritance

[object](#) ← ListExtensions

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## Move<T>(List<T>, int, int)

Moves an item within a [List<T>](#) from one index to another.

```
public static void Move<T>(this List<T> list, int oldIndex, int newIndex)
```

### Parameters

**list** [List](#)<T>

The list in which the item will be moved.

**oldIndex** [int](#)

The current index of the item.

**newIndex** [int](#)

The target index to move the item to.

### Type Parameters

T

The type of elements in the list.

## Remarks

If `oldIndex` and `newIndex` are equal, or if either index is out of range, the method does nothing.

# Class ToleranceGroup<TKey, TElement>

Namespace: [MBFWpfToolkit.Extensions](#)

Assembly: MBFWpfToolkit.dll

Represents a collection of elements that share a common key, with additional functionality for grouping and list operations.

```
public class ToleranceGroup<TKey, TElement> : List<TElement>, IList<TElement>,  
ICollection<TElement>, IList, ICollection, IReadOnlyList<TElement>,  
IReadOnlyCollection<TElement>, IGrouping<TKey, TElement>, IEnumerable<TElement>, IEnumerable
```

## Type Parameters

### TKey

The type of the key associated with the group.

### TElement

The type of the elements in the group.

## Inheritance

[object](#) ← [List](#)<TElement> ← ToleranceGroup<TKey, TElement>

## Implements

[IList](#)<TElement>, [ICollection](#)<TElement>, [IList](#), [ICollection](#),  [IReadOnlyList](#)<TElement>,  
 [IReadOnlyCollection](#)<TElement>, [IGrouping](#)<TKey, TElement>, [IEnumerable](#)<TElement>,  
 [IEnumerable](#)

## Inherited Members

[List](#)<TElement>.Add(TElement), [List](#)<TElement>.AddRange(IEnumerable<TElement>),  
[List](#)<TElement>.AsReadOnly(),  
[List](#)<TElement>.BinarySearch(int, int, TElement, IComparer<TElement>),  
[List](#)<TElement>.BinarySearch(TElement),  
[List](#)<TElement>.BinarySearch(TElement, IComparer<TElement>), [List](#)<TElement>.Clear(),  
[List](#)<TElement>.Contains(TElement),  
[List](#)<TElement>.ConvertAll<TOutput>(Converter<TElement, TOutput>),  
[List](#)<TElement>.CopyTo(TElement[]), [List](#)<TElement>.CopyTo(int, TElement[], int, int),  
[List](#)<TElement>.CopyTo(TElement[], int), [List](#)<TElement>.Exists(Predicate<TElement>),  
[List](#)<TElement>.Find(Predicate<TElement>), [List](#)<TElement>.FindAll(Predicate<TElement>),

[List<TElement>.FindIndex\(Predicate<TElement>\)](#) ,  
[List<TElement>.FindIndex\(int, Predicate<TElement>\)](#) ,  
[List<TElement>.FindIndex\(int, int, Predicate<TElement>\)](#) ,  
[List<TElement>.FindLast\(Predicate<TElement>\)](#) ,  
[List<TElement>.FindLastIndex\(Predicate<TElement>\)](#) ,  
[List<TElement>.FindLastIndex\(int, Predicate<TElement>\)](#) ,  
[List<TElement>.FindLastIndex\(int, int, Predicate<TElement>\)](#) ,  
[List<TElement>.ForEach\(Action<TElement>\)](#) , [List<TElement>.GetEnumerator\(\)](#) ,  
[List<TElement>.GetRange\(int, int\)](#) , [List<TElement>.IndexOf\(TElement\)](#) ,  
[List<TElement>.IndexOf\(TElement, int\)](#) , [List<TElement>.IndexOf\(TElement, int, int\)](#) ,  
[List<TElement>.Insert\(int, TElement\)](#) , [List<TElement>.InsertRange\(int, IEnumerable<TElement>\)](#) ,  
[List<TElement>.LastIndexOf\(TElement\)](#) , [List<TElement>.LastIndexOf\(TElement, int\)](#) ,  
[List<TElement>.LastIndexOf\(TElement, int, int\)](#) , [List<TElement>.Remove\(TElement\)](#) ,  
[List<TElement>.RemoveAll\(Predicate<TElement>\)](#) , [List<TElement>.RemoveAt\(int\)](#) ,  
[List<TElement>.RemoveRange\(int, int\)](#) , [List<TElement>.Reverse\(\)](#) ,  
[List<TElement>.Reverse\(int, int\)](#) , [List<TElement>.Sort\(\)](#) ,  
[List<TElement>.Sort\(IComparer<TElement>\)](#) , [List<TElement>.Sort\(int, int, IComparer<TElement>\)](#) ,  
[List<TElement>.Sort\(Comparison<TElement>\)](#) , [List<TElement>.ToArray\(\)](#) ,  
[List<TElement>.TrimExcess\(\)](#) , [List<TElement>.TrueForAll\(Predicate<TElement>\)](#) ,  
[List<TElement>.Capacity](#) , [List<TElement>.Count](#) , [List<TElement>.this\[int\]](#) , [object.ToString\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Extension Methods

[CollectionExtensions.GroupByWithTolerance<T>\(IEnumerable<T>, Func<T, double>, double\)](#) ,  
[CollectionExtensions.ToObservableCollection<T>\(IEnumerable<T>\)](#) ,  
[ListExtensions.Move<T>\(List<T>, int, int\)](#).

## Remarks

This class combines the functionality of [IGrouping< TKey, TElement >](#) and [List< T >](#), allowing grouped elements to be accessed and manipulated as a list.

## Constructors

### ToleranceGroup(TKey)

Represents a group of items that share a common key and are grouped based on a specified tolerance.

```
public ToleranceGroup(TKey key)
```

## Parameters

### key TKey

The key that identifies the group. Cannot be null.

## Remarks

This class is typically used in scenarios where items need to be grouped by a key with a tolerance-based condition.

# Properties

## Key

Gets the key associated with the current object.

```
public TKey Key { get; }
```

## Property Value

TKey

# Namespace MBFWpfToolkit.Helpers

## Classes

### [MbNotification](#)

Provides utility methods to display styled notifications using MicroBIM.UI's MicroBIM.UI.Controls.Growl system.

### [MbObservableCollection<T>](#)

Represents a collection of objects that supports data binding, notifications, and extended behaviors such as adding or removing multiple items at once.

### [MbResourceHelper](#)

Provides helper methods for retrieving theme and skin resources in a WPF application.

### [PasswordHelper](#)

Provides attached properties and helper methods to enable binding for the Password property of PasswordBox controls.

### [SilentValidator](#)

### [WindowController](#)

Manages the lifecycle of modeless WPF windows to prevent duplicates, maintain a global reference list, and enable programmatic control.

# Class MbNotification

Namespace: [MBFWpfToolkit.Helpers](#)

Assembly: MBFWpfToolkit.dll

Provides utility methods to display styled notifications using MicroBIM.UI's MicroBIM.UI.Controls.Growl system.

```
public static class MbNotification
```

## Inheritance

[object](#) ← MbNotification

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Properties

## Host

```
public static MbWindow Host { get; set; }
```

## Property Value

[MbWindow](#)

# Methods

## ShowError(string, int)

Displays an error notification with a red cross icon.

```
public static void ShowError(string message, int waitTime = 2)
```

## Parameters

## **message** [string](#)

The message to be shown in the notification.

## **waitTime** [int](#)

The duration (in seconds) the notification remains visible. Default is 2 seconds.

# ShowInfo(string, int)

Displays an informational notification with a blue info icon.

```
public static void ShowInfo(string message, int waitTime = 2)
```

## Parameters

### **message** [string](#)

The message to be shown in the notification.

### **waitTime** [int](#)

The duration (in seconds) the notification remains visible. Default is 2 seconds.

# ShowSuccess(string, int)

Displays a success notification with a green check icon.

```
public static void ShowSuccess(string message, int waitTime = 2)
```

## Parameters

### **message** [string](#)

The message to be shown in the notification.

### **waitTime** [int](#)

The duration (in seconds) the notification remains visible. Default is 2 seconds.

## ShowWarning(string, int)

Displays a warning notification with a specified message and optional wait time.

```
public static void ShowWarning(string message, int waitTime = 2)
```

### Parameters

**message** [string](#)

The warning message to display.

**waitTime** [int](#)

The time in seconds to display the notification before it disappears. Defaults to 2 seconds if not specified.

# Class MbObservableCollection<T>

Namespace: [MBFWpfToolkit.Helpers](#)

Assembly: MBFWpfToolkit.dll

Represents a collection of objects that supports data binding, notifications, and extended behaviors such as adding or removing multiple items at once.

```
public class MbObservableCollection<T> : ObservableCollection<T>, IList<T>, ICollection<T>,  
IList, ICollection, IReadOnlyList<T>, IReadOnlyCollection<T>, IEnumerable<T>, I Enumerable,  
INotifyCollectionChanged, INotifyPropertyChanged
```

## Type Parameters

T

The type of elements in the collection.

## Inheritance

[object](#) ← [Collection](#)<T> ← [ObservableCollection](#)<T> ← MbObservableCollection<T>

## Implements

[IList](#)<T>, [ICollection](#)<T>, [IList](#), [ICollection](#),  [IReadOnlyList](#)<T>,  [IReadOnlyCollection](#)<T>,  
[IEnumerable](#)<T>, [IEnumerable](#), [INotifyCollectionChanged](#), [INotifyPropertyChanged](#)

## Inherited Members

[ObservableCollection](#)<T>.Move(int, int), [ObservableCollection](#)<T>.ClearItems(),  
[ObservableCollection](#)<T>.RemoveItem(int), [ObservableCollection](#)<T>.InsertItem(int, T),  
[ObservableCollection](#)<T>.SetItem(int, T), [ObservableCollection](#)<T>.MoveItem(int, int),  
[ObservableCollection](#)<T>.OnPropertyChanged(PropertyChangedEventArgs),  
[ObservableCollection](#)<T>.BlockReentrancy(), [ObservableCollection](#)<T>.CheckReentrancy(),  
[ObservableCollection](#)<T>.CollectionChanged, [ObservableCollection](#)<T>.PropertyChanged,  
[Collection](#)<T>.Add(T), [Collection](#)<T>.Clear(), [Collection](#)<T>.CopyTo(T[], int),  
[Collection](#)<T>.Contains(T), [Collection](#)<T>.GetEnumerator(), [Collection](#)<T>.IndexOf(T),  
[Collection](#)<T>.Insert(int, T), [Collection](#)<T>.Remove(T), [Collection](#)<T>.RemoveAt(int),  
[Collection](#)<T>.Count, [Collection](#)<T>.Items, [Collection](#)<T>.this[int], [object](#).ToString(),  
[object](#).Equals([object](#)), [object](#).Equals([object](#), [object](#)), [object](#).ReferenceEquals([object](#), [object](#)),  
[object](#).GetHashCode(), [object](#).GetType(), [object](#).MemberwiseClone()

## Extension Methods

[CollectionExtensions.GroupByWithTolerance<T>\(IEnumerable<T>, Func<T, double>, double\)](#) ,  
[CollectionExtensions.ToObservableCollection<T>\(IEnumerable<T>\)](#)

## Remarks

This class extends [ObservableCollection<T>](#) to provide additional functionality, including methods for adding and removing multiple items in a single operation while suppressing intermediate notifications. Notifications are raised only once after the operation completes.

## Constructors

### MbObservableCollection()

Initializes a new instance of the [MbObservableCollection<T>](#) class.

```
public MbObservableCollection()
```

### MbObservableCollection(IEnumerable<T>)

Initializes a new instance of the [MbObservableCollection<T>](#) class with the elements copied from the specified collection.

```
public MbObservableCollection(IEnumerable<T> collection)
```

#### Parameters

**collection** [IEnumerable<T>](#)

The collection whose elements are copied to the new [MbObservableCollection<T>](#).

### MbObservableCollection(List<T>)

Initializes a new instance of the [MbObservableCollection<T>](#) class with the specified list as the initial collection of items.

```
public MbObservableCollection(List<T> list)
```

#### Parameters

`list` [List](#)<T>

The list of items to initialize the collection with. Cannot be null.

## Remarks

The provided list is used to populate the initial items in the collection. Changes to the list after initialization will not affect the collection.

# Methods

## MbAddRange(IList<T>)

Adds a range of items to the collection, ensuring that only unique items are added.

```
public void MbAddRange(IList<T> items)
```

### Parameters

`items` [IList](#)<T>

The list of items to add to the collection. Items that already exist in the collection will be ignored.

## Remarks

If `items` is [null](#), the method does nothing. This method suppresses collection change notifications during the addition process and raises a single reset notification after all items are added.

## MbRemoveRange(IEnumerable<T>)

Removes a range of items from the collection.

```
public void MbRemoveRange(IEnumerable<T> items)
```

### Parameters

`items` [IEnumerable](#)<T>

The collection of items to remove. If `items` is [null](#), the method does nothing.

## Remarks

This method suppresses notifications during the removal of items and raises a single [NotifyCollectionChangedEventArgs](#) event with the [Reset](#) action after all items have been removed.

## OnCollectionChanged(NotifyCollectionChangedEventArgs)

Raises the [CollectionChanged](#) event.

```
protected override void OnCollectionChanged(NotifyCollectionChangedEventArgs e)
```

## Parameters

e [NotifyCollectionChangedEventArgs](#)

The event data containing information about the change to the collection.

## Remarks

This method overrides the base implementation to conditionally suppress the notification based on the internal state of the collection. If notifications are not suppressed, the base implementation is invoked to raise the event.

# Class MbResourceHelper

Namespace: [MBFWpfToolkit.Helpers](#)

Assembly: MBFWpfToolkit.dll

Provides helper methods for retrieving theme and skin resources in a WPF application.

```
public class MbResourceHelper
```

## Inheritance

[object](#) ← MbResourceHelper

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Constructors

### MbResourceHelper()

```
public MbResourceHelper()
```

## Methods

### GetResource<T>(string)

Retrieves a resource from the application's merged resource dictionaries by key.

```
public static T GetResource<T>(string key)
```

#### Parameters

key [string](#)

The key of the resource to retrieve.

Returns

T

The resource cast to type T if found; otherwise, the default value of T.

Type Parameters

T

The expected type of the resource.

## GetSkin(SkinType)

Loads a built-in MicroBIM.UI skin resource dictionary by skin type.

```
public static ResourceDictionary GetSkin(SkinType skin)
```

Parameters

skin [SkinType](#)

The [SkinType](#) indicating which skin to load.

Returns

[ResourceDictionary](#)

The loaded [ResourceDictionary](#).

## GetSkin(Assembly, string, SkinType)

Loads a skin resource dictionary from a specified assembly and path.

```
public static ResourceDictionary GetSkin(Assembly assembly, string themePath, SkinType skin)
```

Parameters

assembly [Assembly](#)

The assembly containing the skin resources.

**themePath** [string](#)

The path to the skin directory inside the assembly.

**skin** [SkinType](#)

The [SkinType](#) indicating which skin to load.

Returns

[ResourceDictionary](#)

The loaded [ResourceDictionary](#) for the specified skin.

## GetStandaloneTheme()

Loads the standalone MicroBIM.UI theme resource dictionary.

```
public static ResourceDictionary GetStandaloneTheme()
```

Returns

[ResourceDictionary](#)

The loaded [ResourceDictionary](#).

## GetTheme()

Retrieves the current theme resource dictionary. If not already loaded, it loads the standalone MicroBIM.UI theme.

```
public static ResourceDictionary GetTheme()
```

Returns

[ResourceDictionary](#)

The [ResourceDictionary](#) representing the current theme.

# Class PasswordHelper

Namespace: [MBFWpfToolkit.Helpers](#)

Assembly: MBFWpfToolkit.dll

Provides attached properties and helper methods to enable binding for the Password property of PasswordBox controls.

```
public static class PasswordHelper
```

## Inheritance

[object](#) ← PasswordHelper

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Fields

### AttachProperty

Attached property to enable or disable binding functionality for PasswordBox.

```
public static readonly DependencyProperty AttachProperty
```

## Field Value

[DependencyProperty](#)

### PasswordProperty

Attached property to bind the password value.

```
public static readonly DependencyProperty PasswordProperty
```

## Field Value

## Methods

### GetAttach(DependencyObject)

Gets the value of the Attach property.

```
public static bool GetAttach(DependencyObject dp)
```

Parameters

dp [DependencyObject](#)

Returns

[bool](#)

### GetPassword(DependencyObject)

Gets the bound password string from the attached property.

```
public static string GetPassword(DependencyObject dp)
```

Parameters

dp [DependencyObject](#)

Returns

[string](#)

### SetAttach(DependencyObject, bool)

Sets the Attach property to enable password binding on the given object.

```
public static void SetAttach(DependencyObject dp, bool value)
```

## Parameters

dp [DependencyObject](#)

value [bool](#)

## SetPassword(DependencyObject, string)

Sets the bound password string on the attached property.

```
public static void SetPassword(DependencyObject dp, string value)
```

## Parameters

dp [DependencyObject](#)

value [string](#)

# Class SilentValidator

Namespace: [MBFWpfToolkit.Helpers](#)

Assembly: MBFWpfToolkit.dll

```
public static class SilentValidator
```

## Inheritance

[object](#) ← SilentValidator

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## ValidateAllProperties(object)

Validates all properties of the specified object instance based on data annotation attributes.

```
public static bool ValidateAllProperties(object instance)
```

### Parameters

instance [object](#)

The object instance to validate. Cannot be [null](#).

### Returns

[bool](#)

[true](#) if all properties of the object are valid; otherwise, [false](#).

### Remarks

This method uses data annotation attributes applied to the object's properties to perform validation. If any property fails validation, the method returns [false](#). Note that this method does not provide

detailed validation results; it only indicates overall validity.

## ValidateAllProperties(object, out Dictionary<string, List<string>>)

Validates all properties of the specified object instance using data annotations.

```
public static bool ValidateAllProperties(object instance, out Dictionary<string, List<string>> errors)
```

### Parameters

**instance** [object](#)

The object instance to validate. Cannot be [null](#).

**errors** [Dictionary](#)<[string](#), [List](#)<[string](#)>>

When this method returns, contains a dictionary of validation errors, where the keys are property names and the values are lists of error messages for each property. If the object is valid, this will be an empty dictionary.

### Returns

[bool](#)

[true](#) if the object is valid and all properties pass validation; otherwise, [false](#).

### Remarks

This method uses the [Validator](#) class to perform validation. It validates all properties of the object, including those marked with validation attributes such as [RequiredAttribute](#).

## ValidateProperty(object, string)

Validates the specified property of an object instance against its data annotations.

```
public static bool ValidateProperty(object instance, string propertyName)
```

### Parameters

## **instance** [object](#)

The object instance containing the property to validate. Cannot be [null](#).

## **propertyName** [string](#)

The name of the property to validate. Cannot be [null](#) or empty.

## Returns

### [bool](#)

[true](#) if the property value is valid according to its data annotations; otherwise, [false](#).

## Remarks

This method uses the [TryValidateProperty\(object, ValidationContext, ICollection<ValidationResult>\)](#) method to validate the property value against the data annotations defined on the property. If the property does not exist or is not accessible, an exception will be thrown.

## ValidateProperty(object, string, out List<string>)

Validates the specified property of an object instance against its data annotations.

```
public static bool ValidateProperty(object instance, string propertyName, out  
List<string> errors)
```

## Parameters

### **instance** [object](#)

The object instance containing the property to validate. Cannot be [null](#).

### **propertyName** [string](#)

The name of the property to validate. Cannot be [null](#) or empty.

### **errors** [List](#)<[string](#)>

When the method returns, contains a list of validation error messages if the property is invalid; otherwise, an empty list. This parameter is passed uninitialized.

## Returns

[bool](#)

[true](#) if the property is valid according to its data annotations; otherwise, [false](#).

## Remarks

This method uses the [Validator](#) class to validate the property. The validation is performed based on the data annotations applied to the property.

# Class WindowController

Namespace: [MBFWpfToolkit.Helpers](#)

Assembly: MBFWpfToolkit.dll

Manages the lifecycle of modeless WPF windows to prevent duplicates, maintain a global reference list, and enable programmatic control.

```
public static class WindowController
```

## Inheritance

[object](#) ← WindowController

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## Close<T>()

Closes all windows of type **T** and removes them from the tracking list.

```
public static void Close<T>() where T : Window
```

### Type Parameters

**T**

The type of the window to close.

## Focus<T>()

Attempts to focus an existing window of type **T**. If minimized or hidden, restores and shows it.

```
public static bool Focus<T>() where T : Window
```

Returns

bool ↗

True if an existing window was found and focused; otherwise false.

Type Parameters

T

The type of the window to focus.

**Hide<T>()**

Hides all windows of type T.

```
public static void Hide<T>() where T : Window
```

Type Parameters

T

The type of the window to hide.

**Show(Window)**

Shows a modeless window and registers it in the controller.

```
public static void Show(Window window)
```

Parameters

window Window

The window to show.

**Show(Window, IntPtr)**

Shows a modeless window with a native owner handle.

```
public static void Show(Window window, IntPtr handle)
```

## Parameters

**window** Window

The window to show.

**handle** [IntPtr](#)

The native owner window handle (HWND).

## ShowDialog(Window)

Shows a modal dialog window and registers it in the controller.

```
public static bool? ShowDialog(Window window)
```

## Parameters

**window** Window

The dialog window to show.

## Returns

[bool](#)?

True if the user accepted the dialog (DialogResult == true); otherwise, false.

## ShowDialog(Window, IntPtr)

Displays the specified MicroBIM.UI.Controls.Window as a modal dialog and returns the result.

```
public static bool? ShowDialog(Window window, IntPtr handle)
```

## Parameters

**window** Window

The MicroBIM.UI.Controls.Window to display as a modal dialog. Cannot be null.

#### handle [IntPtr](#)

A handle to the owner window. This establishes the ownership relationship between the dialog and the parent window.

Returns

#### [bool](#)?

A nullable [bool](#) indicating the dialog result: [true](#) if the user accepted the dialog, [false](#) if the user canceled it, or [null](#) if no result was specified.

Remarks

This method sets the owner of the specified MicroBIM.UI.Controls.Window using the provided handle before displaying it. Ensure that the [handle](#) corresponds to a valid window handle to avoid unexpected behavior.

## Show<T>()

Shows an already existing window of type [T](#) if found.

```
public static void Show<T>() where T : Window
```

Type Parameters

#### [T](#)

The type of the window to show.

# Namespace MBFWpfToolkit.MbViewModel

## Classes

### [MbBaseViewModel](#)

Represents the base ViewModel class inheriting from ObservableValidator. Provides a common set of functionalities for ViewModels, including validation capabilities.

# Class MbBaseViewModel

Namespace: [MBFWpfToolkit.MbViewModel](#)

Assembly: MBFWpfToolkit.dll

Represents the base ViewModel class inheriting from ObservableValidator. Provides a common set of functionalities for ViewModels, including validation capabilities.

```
public class MbBaseViewModel : ObservableValidator
```

## Inheritance

[object](#) ← MbBaseViewModel

## Constructors

### MbBaseViewModel()

```
public MbBaseViewModel()
```

## Properties

### MbWindow

Represents a property of type [MbWindow](#) used within the ViewModel.

```
public MbWindow? MbWindow { get; set; }
```

### Property Value

#### [MbWindow](#)

A reference to an instance of [MbWindow](#). This property can be used to set or retrieve details of the associated [MbWindow](#), enabling interaction with its properties, such as theme toggling, supported languages, plugin information, and more.

### Remarks

This property is primarily used to reference an instance of the [MbWindow](#) class, which provides properties and functionality for managing window-level settings such as themes, language settings, and display options for plugins.

# Namespace MBFWpfToolkit.Models

## Classes

### [HelpStep](#)

Represents a help step in the UI help wizard with an associated UI element, descriptive text, and order.

# Class HelpStep

Namespace: [MBFWpfToolkit.Models](#)

Assembly: MBFWpfToolkit.dll

Represents a help step in the UI help wizard with an associated UI element, descriptive text, and order.

```
public class HelpStep
```

## Inheritance

[object](#) ← HelpStep

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Constructors

## HelpStep()

```
public HelpStep()
```

# Properties

## Element

Gets or sets the UI element associated with this help step.

```
public UIElement Element { get; set; }
```

## Property Value

[UIElement](#)

## Remarks

This UI element is used to anchor tooltips or visual guidance during the help walkthrough.

## StepNumber

Gets or sets the order number of the step in the help sequence.

```
public int StepNumber { get; set; }
```

Property Value

[int](#)

## StepText

Gets or sets the textual description of the step to be shown to the user.

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

Property Value

[string](#)

# Namespace MBFWpfToolkit.Properties.Langs

## Classes

### [Lang](#)

A strongly-typed resource class, for looking up localized strings, etc.

# Class Lang

Namespace: [MBFWpfToolkit.Properties.Langs](#)

Assembly: MBFWpfToolkit.dll

A strongly-typed resource class, for looking up localized strings, etc.

```
public class Lang
```

## Inheritance

[object](#) ← Lang

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Properties

## About

Looks up a localized string similar to About.

```
public static string About { get; }
```

## Property Value

[string](#)

## And

Looks up a localized string similar to and.

```
public static string And { get; }
```

## Property Value

[string](#)

## BIMAssets

Looks up a localized string similar to BIMAssets.

```
public static string BIMAssets { get; }
```

Property Value

[string](#)

## BookMode

Looks up a localized string similar to Book Mode.

```
public static string BookMode { get; }
```

Property Value

[string](#)

## Cancel

Looks up a localized string similar to Cancel.

```
public static string Cancel { get; }
```

Property Value

[string](#)

## ClosePDF

Looks up a localized string similar to Close PDF.

```
public static string ClosePDF { get; }
```

Property Value

[string](#)

## Confirm

Looks up a localized string similar to Confirm.

```
public static string Confirm { get; }
```

Property Value

[string](#) ↗

## ContinuousPagesMode

Looks up a localized string similar to Continuous Pages Mode.

```
public static string ContinuousPagesMode { get; }
```

Property Value

[string](#) ↗

## Culture

Overrides the current thread's CurrentUICulture property for all resource lookups using this strongly typed resource class.

```
public static CultureInfo Culture { get; set; }
```

Property Value

[CultureInfo](#) ↗

## DPI\_

Looks up a localized string similar to DPI:.

```
public static string DPI_ { get; }
```

## PropertyValue

[string ↗](#)

## DecimalValue

Looks up a localized string similar to Value must be a valid decimal..

```
public static string DecimalValue { get; }
```

## PropertyValue

[string ↗](#)

## Delete

Looks up a localized string similar to Delete.

```
public static string Delete { get; }
```

## PropertyValue

[string ↗](#)

## DoubleValue

Looks up a localized string similar to Value must be a valid double..

```
public static string DoubleValue { get; }
```

## PropertyValue

[string ↗](#)

## Download

Looks up a localized string similar to Download.

```
public static string Download { get; }
```

Property Value

[string](#) ↗

## DownloadSelected

Looks up a localized string similar to DownloadSelected.

```
public static string DownloadSelected { get; }
```

Property Value

[string](#) ↗

## Edit

Looks up a localized string similar to Edit.

```
public static string Edit { get; }
```

Property Value

[string](#) ↗

## FileType

Looks up a localized string similar to File Type.

```
public static string FileType { get; }
```

Property Value

[string](#) ↗

## FileURL

Looks up a localized string similar to File URL.

```
public static string FileURL { get; }
```

Property Value

[string](#)

## Findindocument

Looks up a localized string similar to Find in document....

```
public static string Findindocument { get; }
```

Property Value

[string](#)

## FitHeight

Looks up a localized string similar to Fit Height.

```
public static string FitHeight { get; }
```

Property Value

[string](#)

## FitWidth

Looks up a localized string similar to Fit Width.

```
public static string FitWidth { get; }
```

Property Value

[string](#)

## From

Looks up a localized string similar to From.

```
public static string From { get; }
```

Property Value

[string](#) ↗

## GetPageText

Looks up a localized string similar to Get Page Text.

```
public static string GetPageText { get; }
```

Property Value

[string](#) ↗

## HandTool

Looks up a localized string similar to Hand Tool.

```
public static string HandTool { get; }
```

Property Value

[string](#) ↗

## Installers

Looks up a localized string similar to Installers.

```
public static string Installers { get; }
```

Property Value

[string](#) ↗

## IntValue

Looks up a localized string similar to Value must be a valid integer..

```
public static string IntValue { get; }
```

Property Value

[string](#) ↗

## InvalidFormat

Looks up a localized string similar to InvalidFormat!.

```
public static string InvalidFormat { get; }
```

Property Value

[string](#) ↗

## Islefttorightdocument

Looks up a localized string similar to Is left to right document.

```
public static string Islefttorightdocument { get; }
```

Property Value

[string](#) ↗

## Isrighttoleftdocument

Looks up a localized string similar to Is right to left document.

```
public static string Isrighttoleftdocument { get; }
```

Property Value

[string](#)

## Light\_DarkMode

Looks up a localized string similar to Light/Dark Mode.

```
public static string Light_DarkMode { get; }
```

Property Value

[string](#)

## Miscellaneous

Looks up a localized string similar to Miscellaneous.

```
public static string Miscellaneous { get; }
```

Property Value

[string](#)

## NextPage

Looks up a localized string similar to Next Page.

```
public static string NextPage { get; }
```

Property Value

[string](#)

## No

Looks up a localized string similar to No.

```
public static string No { get; }
```

## PropertyValue

[string](#)

## NumberRange

Looks up a localized string similar to Value must be between.

```
public static string NumberRange { get; }
```

## PropertyValue

[string](#)

## OpenPdf

Looks up a localized string similar to Open Pdf.

```
public static string OpenPdf { get; }
```

## PropertyValue

[string](#)

## PdfInformation

Looks up a localized string similar to Pdf Information.

```
public static string PdfInformation { get; }
```

## PropertyValue

[string](#)

## PreviousPage

Looks up a localized string similar to Previous Page.

```
public static string PreviousPage { get; }
```

Property Value

[string](#) ↗

## Renderallpages

Looks up a localized string similar to Render all pages.

```
public static string Renderallpages { get; }
```

Property Value

[string](#) ↗

## Required

Looks up a localized string similar to Value cannot be empty..

```
public static string Required { get; }
```

Property Value

[string](#) ↗

## ResourceManager

Returns the cached ResourceManager instance used by this class.

```
public static ResourceManager ResourceManager { get; }
```

## Property Value

[ResourceManager](#) ↗

## RotateLeft

Looks up a localized string similar to Rotate Left.

```
public static string RotateLeft { get; }
```

## Property Value

[string](#) ↗

## RotateRight

Looks up a localized string similar to Rotate Right.

```
public static string RotateRight { get; }
```

## Property Value

[string](#) ↗

## Save

Looks up a localized string similar to Save.

```
public static string Save { get; }
```

## Property Value

[string](#) ↗

## SaveasImages

Looks up a localized string similar to Save as Images.

```
public static string SaveasImages { get; }
```

Property Value

[string](#)

## Searchforinstallers

Looks up a localized string similar to Search for installers....

```
public static string Searchforinstallers { get; }
```

Property Value

[string](#)

## Searchterm

Looks up a localized string similar to Search term.

```
public static string Searchterm { get; }
```

Property Value

[string](#)

## SelectLanguage

Looks up a localized string similar to Select Language.

```
public static string SelectLanguage { get; }
```

Property Value

[string](#)

## ShowBookmarks

Looks up a localized string similar to Show Bookmarks.

```
public static string ShowBookmarks { get; }
```

Property Value

[string](#)

## SinglePageMode

Looks up a localized string similar to Single Page Mode.

```
public static string SinglePageMode { get; }
```

Property Value

[string](#)

## SwapToDarkMode

Looks up a localized string similar to Swap to dark mode.

```
public static string SwapToDarkMode { get; }
```

Property Value

[string](#)

## SwapToLightMode

Looks up a localized string similar to Swap to light mode.

```
public static string SwapToLightMode { get; }
```

Property Value

[string](#)

## To

Looks up a localized string similar to To.

```
public static string To { get; }
```

Property Value

[string](#) ↗

## TransparentPage

Looks up a localized string similar to Transparent Page.

```
public static string TransparentPage { get; }
```

Property Value

[string](#) ↗

## Upload

Looks up a localized string similar to Upload.

```
public static string Upload { get; }
```

Property Value

[string](#) ↗

## UploadedDate

Looks up a localized string similar to Uploaded Date.

```
public static string UploadedDate { get; }
```

## Property Value

[string](#) ↗

## Url

Looks up a localized string similar to URL.

```
public static string Url { get; }
```

## Property Value

[string](#) ↗

## Version

Looks up a localized string similar to Version.

```
public static string Version { get; }
```

## Property Value

[string](#) ↗

## Yes

Looks up a localized string similar to Yes.

```
public static string Yes { get; }
```

## Property Value

[string](#) ↗

## Zoomin

Looks up a localized string similar to Zoom in.

```
public static string Zoomin { get; }
```

Property Value

[string ↗](#)

## Zoomout

Looks up a localized string similar to Zoom out.

```
public static string Zoomout { get; }
```

Property Value

[string ↗](#)

## contact\_us

Looks up a localized string similar to Contact us.

```
public static string contact_us { get; }
```

Property Value

[string ↗](#)

## of

Looks up a localized string similar to of.

```
public static string of { get; }
```

Property Value

[string ↗](#)

# Namespace MBFWpfToolkit.Services

## Classes

### [AspectServiceAccessor](#)

Retrieves a service of the specified type using the configured resolver function.

### [DialogService](#)

Provides functionality for displaying, updating, and closing dialogs in an application.

## Interfaces

### [IDialogService](#)

Provides methods for displaying, updating, and closing dialog windows.

### [IUpdatableDialogContent](#)

Represents a dialog content that can be updated dynamically with a new message.

# Class AspectServiceAccessor

Namespace: [MBFWpfToolkit.Services](#)

Assembly: MB.Common.dll

Retrieves a service of the specified type using the configured resolver function.

```
public static class AspectServiceAccessor
```

## Inheritance

[object](#) ← AspectServiceAccessor

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

This method uses the [ServiceResolver](#) delegate to resolve the requested service. Ensure that the resolver function is properly configured to handle the requested type.

## Properties

### ServiceResolver

Gets or sets the function used to resolve services by type.

```
public static Func<Type, object>? ServiceResolver { get; set; }
```

### Property Value

[Func](#)<[Type](#), [object](#)>

## Remarks

This property is typically used to provide a custom service resolution mechanism, such as dependency injection. The delegate should return an instance of the requested service type or throw an exception if the service cannot be resolved.

# Methods

## GetService<T>()

Retrieves an instance of the specified service type.

```
public static T GetService<T>() where T : class
```

Returns

T

An instance of the requested service type T.

Type Parameters

T

The type of the service to retrieve. Must be a reference type.

Remarks

This method uses a service resolver delegate to resolve the requested service. Ensure that `AspectServiceAccessor.ServiceResolver` is properly set before calling this method.

Exceptions

`InvalidOperationException`

Thrown if the service resolver is not configured.

# Class DialogService

Namespace: [MBFWpfToolkit.Services](#)

Assembly: MBFWpfToolkit.dll

Provides functionality for displaying, updating, and closing dialogs in an application.

```
public class DialogService : IDialogService
```

## Inheritance

[object](#) ← DialogService

## Implements

[IDialogService](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Remarks

The [DialogService](#) class manages the lifecycle of dialogs, including showing, updating messages, and closing dialogs. It supports multiple nested dialogs.

## Constructors

### DialogService()

```
public DialogService()
```

## Methods

### Close()

Closes the top-most dialog if it is currently open.

```
public void Close()
```

## CloseAll()

Closes all open dialogs.

```
public void CloseAll()
```

## Show<T>(T)

Displays the specified content in a dialog.

```
public void Show<T>(T content)
```

Parameters

**content** T

The content to display in the dialog. Cannot be null.

Type Parameters

T

The type of the content to display in the dialog.

## UpdateMessage(string)

Updates the message of the top-most open dialog.

```
public void UpdateMessage(string message)
```

Parameters

**message** string ↗

The new message to display. Cannot be null.

# Interface IDialogService

Namespace: [MBFWpfToolkit.Services](#)

Assembly: MBFWpfToolkit.dll

Provides methods for displaying, updating, and closing dialog windows.

```
public interface IDialogService
```

## Remarks

This interface defines a contract for managing dialog windows, including showing content, updating messages, and closing dialogs. Implementations of this interface can be used to interact with user interface dialogs in a consistent manner.

## Methods

### Close()

Closes the current connection or resource, releasing any associated resources.

```
void Close()
```

#### Remarks

Once the connection or resource is closed, it cannot be reused. Ensure that all necessary operations are completed before calling this method.

### Show<T>(T)

Displays the specified content.

```
void Show<T>(T content)
```

#### Parameters

**content** T

The content to be displayed. Cannot be null.

## Type Parameters

T

The type of the content to display.

## UpdateMessage(string)

Updates the current message with the specified value.

```
void UpdateMessage(string message)
```

### Parameters

**message** [string](#)

The new message to set. Cannot be null or empty.

### Remarks

This method replaces the existing message with the provided value. Ensure that **message** is a valid, non-empty string before calling this method.

# Interface IUpdatableDialogContent

Namespace: [MBFWpfToolkit.Services](#)

Assembly: MBFWpfToolkit.dll

Represents a dialog content that can be updated dynamically with a new message.

```
public interface IUpdatableDialogContent
```

## Remarks

This interface is typically used in scenarios where the content of a dialog needs to be updated after it has been displayed, such as progress dialogs or notifications.

## Methods

### UpdateMessage(string)

Updates the current message with the specified value.

```
void UpdateMessage(string message)
```

#### Parameters

**message** [string](#) ↗

The new message to set. Cannot be null or empty.

#### Remarks

This method replaces the existing message with the provided value. Ensure that **message** contains valid content before calling this method.

# Namespace MBFWpfToolkit.Utils

## Classes

### [WindowUtils](#)

Provides utility functions for managing windows and processes in a WPF application.

# Class WindowUtils

Namespace: [MBFWpfToolkit.Utils](#)

Assembly: MBFWpfToolkit.dll

Provides utility functions for managing windows and processes in a WPF application.

```
public static class WindowUtils
```

## Inheritance

[object](#) ← WindowUtils

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### OpenLink(object)

Opens a link in the default web browser or associated application.

```
public static void OpenLink(object url)
```

#### Parameters

`url` [object](#)

The URL or file path to open. This should be a valid string representation of the resource.

#### Exceptions

[ArgumentNullException](#)

Thrown when the `url` is null.

[InvalidOperationException](#)

Thrown when the `url` is not a valid string.

# Namespace MBFWpfToolkit.ValidationRules

## Classes

### [DecimalValidationRule](#)

Validation rule for decimal input with localization support.

### [DoubleValidationRule](#)

Validation rule for double input with localization support.

### [IntegerValidationRule](#)

Validation rule for integer input with localization support.

### [StringValidationRule](#)

Validation rule for string input with localization support.

# Class DecimalValidationRule

Namespace: [MBFWpfToolkit.ValidationRules](#)

Assembly: MBFWpfToolkit.dll

Validation rule for decimal input with localization support.

```
public class DecimalValidationRule : ValidationRule
```

## Inheritance

[object](#) ← [ValidationRule](#) ← DecimalValidationRule

## Inherited Members

[ValidationRule.Validate\(object, CultureInfo, BindingExpressionBase\)](#) ,  
[ValidationRule.Validate\(object, CultureInfo, BindingGroup\)](#) , [ValidationRule.ValidationStep](#) ,  
[ValidationRule.ValidatesOnTargetUpdated](#) , [object.ToString\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Constructors

### DecimalValidationRule()

```
public DecimalValidationRule()
```

## Properties

### IsRequired

Gets or sets a value indicating whether the input is required.

```
public bool IsRequired { get; set; }
```

### Property Value

[bool](#)

## Maximum

Gets or sets the maximum allowable value.

```
public decimal Maximum { get; set; }
```

Property Value

[decimal](#)

## Minimum

Gets or sets the minimum allowable value.

```
public decimal Minimum { get; set; }
```

Property Value

[decimal](#)

## Methods

### Validate(object, CultureInfo)

Validates the input value.

```
public override ValidationResult Validate(object value, CultureInfo cultureInfo)
```

Parameters

**value** [object](#)

The value to validate.

**cultureInfo** [CultureInfo](#)

The culture information.

Returns

## ValidationResult

A [ValidationResult](#) indicating whether the value is valid.

# Class DoubleValidationRule

Namespace: [MBFWpfToolkit.ValidationRules](#)

Assembly: MBFWpfToolkit.dll

Validation rule for double input with localization support.

```
public class DoubleValidationRule : ValidationRule
```

## Inheritance

[object](#) ← [ValidationRule](#) ← DoubleValidationRule

## Inherited Members

[ValidationRule.Validate\(object, CultureInfo, BindingExpressionBase\)](#) ,  
[ValidationRule.Validate\(object, CultureInfo, BindingGroup\)](#) , [ValidationRule.ValidationStep](#) ,  
[ValidationRule.ValidatesOnTargetUpdated](#) , [object.ToString\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Constructors

### DoubleValidationRule()

```
public DoubleValidationRule()
```

## Properties

### IsRequired

Gets or sets a value indicating whether the input is required.

```
public bool IsRequired { get; set; }
```

### Property Value

[bool](#)

## Maximum

Gets or sets the maximum allowable value.

```
public double Maximum { get; set; }
```

Property Value

[double](#)

## Minimum

Gets or sets the minimum allowable value.

```
public double Minimum { get; set; }
```

Property Value

[double](#)

## Methods

### Validate(object, CultureInfo)

Validates the input value.

```
public override ValidationResult Validate(object value, CultureInfo cultureInfo)
```

Parameters

**value** [object](#)

The value to validate.

**cultureInfo** [CultureInfo](#)

The culture information.

Returns

## ValidationResult

A [ValidationResult](#) indicating whether the value is valid.

# Class IntegerValidationRule

Namespace: [MBFWpfToolkit.ValidationRules](#)

Assembly: MBFWpfToolkit.dll

Validation rule for integer input with localization support.

```
public class IntegerValidationRule : ValidationRule
```

## Inheritance

[object](#) ← [ValidationRule](#) ← IntegerValidationRule

## Inherited Members

[ValidationRule.Validate\(object, CultureInfo, BindingExpressionBase\)](#) ,  
[ValidationRule.Validate\(object, CultureInfo, BindingGroup\)](#) , [ValidationRule.ValidationStep](#) ,  
[ValidationRule.ValidatesOnTargetUpdated](#) , [object.ToString\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Constructors

### IntegerValidationRule()

```
public IntegerValidationRule()
```

## Properties

### IsRequired

Gets or sets a value indicating whether the input is required.

```
public bool IsRequired { get; set; }
```

### Property Value

[bool](#)

## Maximum

Gets or sets the maximum allowable value.

```
public int Maximum { get; set; }
```

### Property Value

[int](#)

## Minimum

Gets or sets the minimum allowable value.

```
public int Minimum { get; set; }
```

### Property Value

[int](#)

## Methods

### Validate(object, CultureInfo)

Validates the input value.

```
public override ValidationResult Validate(object value, CultureInfo cultureInfo)
```

#### Parameters

**value** [object](#)

The value to validate.

**cultureInfo** [CultureInfo](#)

The culture information.

#### Returns

## ValidationResult

A [ValidationResult](#) indicating whether the value is valid.

# Class StringValidationRule

Namespace: [MBFWpfToolkit.ValidationRules](#)

Assembly: MBFWpfToolkit.dll

Validation rule for string input with localization support.

```
public class StringValidationRule : ValidationRule
```

## Inheritance

[object](#) ← [ValidationRule](#) ← StringValidationRule

## Inherited Members

[ValidationRule.Validate\(object, CultureInfo, BindingExpressionBase\)](#) ,  
[ValidationRule.Validate\(object, CultureInfo, BindingGroup\)](#) , [ValidationRule.ValidationStep](#) ,  
[ValidationRule.ValidatesOnTargetUpdated](#) , [object.ToString\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Constructors

### StringValidationRule()

```
public StringValidationRule()
```

## Properties

### IsRequired

Gets or sets a value indicating whether the input is required.

```
public bool IsRequired { get; set; }
```

### Property Value

[bool](#)

## MaximumLength

Gets or sets the maximum allowable length.

```
public int MaximumLength { get; set; }
```

Property Value

[int](#)

## MinimumLength

Gets or sets the minimum allowable length.

```
public int MinimumLength { get; set; }
```

Property Value

[int](#)

## Pattern

Gets or sets the regular expression pattern for validation.

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

Property Value

[string](#)

## Methods

### Validate(object, CultureInfo)

Validates the input value.

```
public override ValidationResult Validate(object value, CultureInfo cultureInfo)
```

## Parameters

**value** [object](#)

The value to validate.

**cultureInfo** [CultureInfo](#)

The culture information.

## Returns

[ValidationResult](#)

A [ValidationResult](#) indicating whether the value is valid.