Major changes and renovations to the Revit API

1. API changes

1.1. New version of CEFSharp

Revit and Autodesk add-ins use the CEFsharp library internally for several features. Some third-party add-ins do so as well. Occasionally, when different versions of the library are used, it leads to instability issues for Revit. In order to avoid version conflicts, we are clarifying what CEFsharp version is being used, and loading it prior to all add-in initializations.

- In this version, Revit uses CEFsharp version 92.0.260.

1.2. Replacement for Structural Analytical Model elements

The elements used for the Structural Analytical Model have been completely replaced and the interactions overhauled in this version of Revit. Because of these changes, many pre-existing API classes and methods have been removed immediately in this release as deprecation was not possible. The classes that are impacted are all in the Autodesk.Revit.DB.Structure namespace.

1.2.1. Analytical Model classes

Removed API	Replacement
AnalyticalModel	AnalyticalElement
AnalyticalModelStick	AnalyticalMember
AnalyticalModelColumn	AnalyticalMember
AnalyticalModelSupport	None
AnalyticalConsistencyChecking	None
AnalyticalModelSketchComponent	None
AnalyticalModelSurface	AnalyticalPanel

1.2.2. Analytical Models classes additions

This new API is part of a broader Analytical Driven Modeling initiative for Revit that introduces a new approach to analytical modelling, enhancing its overall structural modelling capabilities. See the RevitAPI.chm for a full listing of new functions.

New classes include:

- Autodesk.Revit.DB.Structure.AnalyticalElement - Represents the analytical portion of a given structural element.
- Autodesk.Revit.DB.Structure.AnalyticalMember - Represents a linear element in the structural analytical model.
- Autodesk.Revit.DB.Structure.AnalyticalPanel - Represents a surface in the structural analytical model.
- Autodesk.Revit.DB.Structure.AnalyticalToPhysicalAssociationManager - Manages the relationship between an analytical element and its physical (model) counterpart.
- Autodesk.Revit.DB.Structure.LevelAssociationData - Holds the information related to level association.
- Autodesk.Revit.DB.Structure.AnalyticalOpening - Represents an opening in an analytical panel element.
- Autodesk.Revit.DB.Structure.ReleaseConditions - Represents a release condition on an analytical element.
- Autodesk.Revit.DB.Structure.AnalyticalNodeData - Holds information related to the analytical model.

1.2.3. Analytical Model enums

Removed API
AnalyticalCurveType
AnalyticalDirection
AnalyticalProjectionType

1.2.4. Element

Removed API	Replacement
Element.GetAnalyticalModel	AnalyticalToPhysicalRelationManager.GetCounterpartsIds
Element.GetAnalyticalModelId	AnalyticalToPhysicalRelationManager.GetCounterpartsIds

1.2.5. StructuralSettings

The following properties have been removed:

Removed API
AnalyticalModelAutoCheckMemberSupports
AnalyticalModelAutoCheckConsistency
AnalyticalModelCheckSupportDistance
AnalyticalModelCheckBeamSlabDistance

Removed API
AnalyticalModelCheckInstability
DifferentiateAnalyticalEnds
AnalyticalModelCheckAdjustment
CheckAnalyticalModelAsset
AnalyticalModelSupportDistanceTolerance
AnalyticalModelDiscrepancyTolerance
AnalyticalModelHorizontalAutofixTolerance
AnalyticalModelVerticalAutofixTolerance
AnalyticalModelCheckCircularReferences
AnalyticalModelCheckDiscrepancy
AnalyticalLinkAutofixTolerance

1.2.6. Line Load, Area Load, Point Load

Removed API	Replacement
Autodesk.Revit.DB.Structure.LineLoad.Create(Document doc, AnalyticalModelStick host, XYZ forceVector1, XYZ momentVector1, LineLoadType symbol)	Autodesk.Revit.DB.Structure.LineLoad.Create(Document doc, hostElemId, XYZ forceVector1, XYZ momentVector1, LineLoadType symbol)
Autodesk.Revit.DB.Structure.LineLoad.Create(Document doc, AnalyticalModelSurface host, int curveIndex, XYZ forceVector1, XYZ momentVector1, LineLoadType symbol)	Autodesk.Revit.DB.Structure.LineLoad.Create(Document doc, hostElemId, int curveIndex, XYZ forceVector1, XYZ momentVector1, LineLoadType symbol)
Autodesk.Revit.DB.Structure.AreaLoad.Create(Document doc, AnalyticalModelSurface host, XYZ forceVector1, AreaLoadType symbol)	Autodesk.Revit.DB.Structure.AreaLoad.Create(Document doc, hostElemId, XYZ forceVector1, AreaLoadType symbol)
Autodesk.Revit.DB.Structure.PointLoad.Create(Document doc, AnalyticalModelStick host, AnalyticalElementSelector selector, XYZ forceVector, XYZ momentVector, PointLoadType symbol)	Autodesk.Revit.DB.Structure.PointLoad.Create(Document doc, hostId, AnalyticalElementSelector selector, XYZ forceVector, XYZ momentVector, PointLoadType symbol)
Autodesk.Revit.DB.Structure.PointLoad.Create(Document doc, AnalyticalModel host, XYZ forceVector, XYZ momentVector, PointLoadType symbol)	None

1.2.7. Load Base

Removed API	Replacement
Autodesk.Revit.DB.Structure.LoadBase.HostElement	HostElementId

1.2.8. Boundary Conditions

Removed API	Replacement
Autodesk.Revit.DB.Structure.BoundaryConditions.HostElement	HostElementId

1.3. Geometry API changes

1.3.1. Bounding Box

The class BoundingBoxXYZ:

description was updated to mention that the bounding box might not be empty even if the element's Geometry property returns an empty result, because GetBoundingBox does not take the Geometry property's Options into account.

1.3.2. Geometry Instance

The property Element Symbol of the GeometryInstance was deprecated.

The replacement
is GeometryInstance.GetDocument().GetElement(GeometryInstance.GetSymbolGeometryId().SymbolId)

1.4. Custom Export API changes

Deprecated Methods:

GroupNode is the base class for InstanceNode and LinkNode

Deprecated API	Replacement
GroupNode.GetSymbolId()	The replacement for LinkNode is the property LinkNode.SymbolId
	The replacement for InstanceNode is InstanceNode.GetSymbolGeometryId()

1.5. Export API changes

The Enum STLExportResolution has been deprecated and replaced:

Deprecated API	Replacement
STLExportResolution enum	ExportResolution enum

Deprecated API	Replacement
STLExportOptions(STLExportResolution)	STLExportOptions(ExportResolution)
STLExportOptions.SetTessellationSettings(STLExportResolution)	STLExportOptions.SetTessellationSettings(ExportResolution)

1.6. Energy Analysis API changes

The property:

- Autodesk.Revit.DB.Analysis.EnergyAnalysisSpace.Area

is modified to cover all scenarios. Previously, it was only for the spaces created with the mode 'Use Rooms or Spaces', and represented the enclosed area measured by interior bounding surfaces. The additional cases include the spaces created with mode 'Use Building Elements' and 'Use Conceptual Masses and Building Elements', where the area is measured by the center plane of walls and the top plane of roofs and floors.

Deprecated API	Replacement
EnergyAnalysisSpace.InnerVolume	EnergyAnalysisSpace.Volume
EnergyAnalysisSpace.AnalyticalVolume	EnergyAnalysisSpace.Volume

Conceptual Masses

A number of classes and methods related to using Conceptual Masses in Energy Analysis have been deprecated. These are replaced by classes and methods in detailed energy analysis.

Deprecated API	Replacement
DB.Analysis.AnalysisMode.ConceptualMasses	ConceptualMassesAndBuildingElements
DB.Analysis.EnergyDataSettings.setCreateAnalyticalModel()	Use EnergyModel property instead.
DB.Analysis.EnergyDataSettings.enableConceptualEnergyAnalyticalModel()	Use EnergyModel property instead.
DB.Analysis.MassGBXMLExportOptions	GBXMLExportOptions
DB.Analysis.MassEnergyAnalyticalMode	EnergyAnalyticalDetailModel
DB.Analysis.MassZone	EnergyAnalyticalDetailModel and EnergyAnalysisSpace
DB.Analysis.MassLevelData	EnergyAnalyticalDetailModel EnergyAnalysisSpace
DB.Analysis.MassSurfaceData	EnergyAnalyticalDetailModel and EnergyAnalysisSurface

1.7. Reinforcement API changes

1.7.1. Reinforcement Elements

The following methods were deprecated due to the fact that the Reinforcement elements representation as solid in a 3D view will be done automatically on Fine detail level. There are no replacements for these capabilities because the geometry is handled automatically.

Deprecated API
Rebar.IsSolidInView()

Deprecated API
Rebar.SetSolidInView()
AreaReinforcement.IsSolidInView()
AreaReinforcement.SetSolidInView()
PathReinforcement.IsSolidInView()
PathReinforcement.SetSolidInView()
RebarInSystem.IsSolidInView()
RebarInSystem.SetSolidInView()
FabricSheet.IsSolidInView()
FabricSheet.SetSolidInView()
RebarContainer.IsSolidInView()
RebarContainer.SetSolidInView()

1.8. Project Browser API removal

The enum value Autodesk.Revit.DB.BrowserOrganizationType.Families of BrowserOrganizationType will no longer be exposed to the public API, as there was no user-facing way to use the value.

1.9. Obsolete API removal

The following API members and classes which had previously been marked Deprecated have been removed in this release. Consult the API documentation from prior releases for information on the replacements to use:

1.9.1. Classes

- Autodesk.Revit.UI.SetupEnergySimulationDialog

1.9.2. Methods

- AnalyticalToPhysicalRelationManager.UpdateRelation()
- AnalyticalToPhysicalRelationManager.GetNeighborIds()
- Autodesk.Revit.Creation.Document.NewFloor()
- Autodesk.Revit.Creation.Document.NewSlab()
- Autodesk.Revit.Creation.Document.NewFoundationSlab()
- Definition.GetSpecTypeId()
- ExternalDefinitionCreationOptions(string, ParameterType)
- FamilyManager.AddParameter(string, BuiltInParameterGroup, ParameterType, bool)
- FabricationNetworkChangeService.SetGroupId()
- FabricationNetworkChangeService.SetRestrictGroup()
- FabricationService.IsValidGroupIndex()
- FabricationService.GetGroupName()
- FabricationService.IsGroupExcluded()
- FabricationService.SetServiceGroupExclusions()
- GlobalParameter.Create(Document, string, ParameterType)
- GlobalParameter.IsValidDataType(ParameterType)

- IndependentTag.GetTaggedLocalElement()
- IndependentTag.GetTaggedReference()
- LabelUtils.GetLabelFor()
- RevisionSettings.GetNumericRevisionSettings()
- RevisionSettings.SetNumericRevisionSettings()
- RevisionSettings.GetAlphanumericRevisionSettings()
- RevisionSettings.SetAlphanumericRevisionSettings()
- TemperatureRatingType.AddCorrectionFactor()
- UnitUtils.GetAllSpecs()
- UnitUtils.GetUnitGroup()
- UnitUtils.IsSpec()

1.9.3. Properties

- BarTypeDiameterOptions.BarDiameter
- CorrectionFactor.Temperature
- Definition.ParameterType
- ExternalDefinition.ParameterType
- ExternalDefinitionCreationOptions.Type
- FabricationService.GroupCount
- FabricationServiceButton.GroupIndex
- FabricationPartSizeMap.GroupId
- FabricationConfigurationInfo.CloudVersion
- FabricationConfigurationInfo.IsConnected
- IndependentTag.TaggedLocalElementId
- IndependentTag.TaggedElementId
- IndependentTag.HasElbow
- IndependentTag.LeaderElbow
- IndependentTag.LeaderEnd
- InternalDefinition.ParameterType
- RebarBarType.BarDiameter
- RebarBendData.BarDiameter
- RebarUpdateCurvesData.GetBarDiameter
- RevisionSettings.NumberType

1.9.4. Enums

- ParameterType
- UnitGroup

2. API additions

2.1. Schedule API additions

2.1.1. Schedule heights on sheets

The new class:

- Autodesk.Revit.DB.ScheduleHeightsOnSheet

returns the heights of schedule title, column header and each body row on sheet view.

The new method:

- ViewSchedule.GetScheduleHeightsOnSheet()

will return the heights object.

2.1.2. Managing schedule segments

The new methods:

- ViewSchedule.IsSplit()
- ViewSchedule.Split(int segmentNumber)
- ViewSchedule.Split(IList<double> segmentHeights)
- ViewSchedule.SplitSegment()
- ViewSchedule.DeleteSegment()
- ViewSchedule.MergeSegments()
- ViewSchedule.GetSegmentCount()
- ViewSchedule.GetSegmentHeight()
- ViewSchedule.SetSegmentHeight()

provide the ability to split schedules and manage schedule segments.

The new method:

- ViewSchedule.GetScheduleInstances()

will return the schedule sheet instances for a schedule segment.

The new methods:

- ScheduleSheetInstance.SegmentIndex
- ScheduleSheetInstance.Create(Document document, ElementId viewSheetId, ElementId scheduleId, XYZ origin, int segmentIndex)

provide the ability to place a schedule segment on sheet and to get and set the schedule segment instance's segment index.

2.1.3. ScheduleDefinition

The new property:

- Autodesk.Revit.DB.ScheduleDefinition.IsFilteredBySheet

indicates if the schedule is set to filter by sheet.

The new method:

- Autodesk.Revit.DB.ScheduleDefinition.IsValidCategoryForFilterBySheet()

checks whether a schedule can be filtered by sheet.

2.2. Worksharing API additions

2.2.1. Delete Workset API

The new method:

- WorksetTable.DeleteWorkset()

supports deleting of worksets from the model. It takes a DeleteWorksetSettings input with options for what to do with elements contained by that workset.

2.3. Geometry API additions

2.3.1. Bounding Box API

The new property:

- BoundingBoxXYZ.IsSet

checks if the bounding box is Enabled and not empty.

2.3.2. Geometry managed by symbol element

The new class

- Autodesk.Revit.DB.SymbolGeometryId

is used to identify a piece of geometry managed by a symbol element. Can be used to compare if two instances point to the same piece of geometry managed by a symbol element.

The new method:

- SymbolGeometryId.AsUniqueIdentifier()

this will convert the SymbolGeometryId to a string that can be used in comparisons to see if two instances point to the same piece of geometry managed by the same symbol element.

The new property:

- SymbolGeometryId.SymbolId

The id of the symbol that is containing the geometry that is shared.

2.4. DirectShape API additions

2.4.1. Type assignability

The new property:

- DirectShapeType.UserAssignability

provides API-level control for whether a given DirectShapeType will appear in the type selector drop-down in the Revit user interface and whether an existing DirectShape can have its type changed to the given DirectShapeType.

2.5. Import/Export API additions

2.5.1. AXM Import

The new class:

- Autodesk.Revit.DB.AXMImportOptions

allows user to determine the import options when importing an AXM file.

The new method:

- Document.Import(String,AXMImportOptions, View)

imports an AXM file into the document.

The new method:

- OptionalFunctionalityUtils.IsAXMImportLinkAvailable()

checks if the Import FormIt function is available.

2.5.2. OBJ Export

The new method:

- Document.Export(String, String, OBJExportOptions)

supports export of Revit geometry to OBJ format. It uses a new class containing the options available for export:

- OBJExportOptions.TargetUnit
- OBJExportOptions.SurfaceTolerance
- OBJExportOptions.NormalTolerance
- OBJExportOptions.MaxEdgeLength
- OBJExportOptions.GridAspectRatio
- OBJExportOptions.SetTessellationSettings()

2.5.3. STL and OBJ Import & Link

The new methods:

- Document.Import(String, OBJImportOptions, View)
- Document.Import(String, STLImportOptions, View)
- ImportInstance.Create(Document, View, ExternalResourceReference, OBJImportOptions, [out] LinkLoadResult)
- ImportInstance.Create(Document, View, ExternalResourceReference, STLImportOptions, [out] LinkLoadResult)
- Document.Link(String, OBJImportOptions, View)
- Document.Link(String, STLImportOptions, View)

provide support for import and linking of files of STL and OBJ formats. These methods use new classes representing the options for each of the new formats.

2.5.4. ShapeImporter API additions

The new enumerations:

- ShapeImporter.SKP
- ShapeImporter.STL
- ShapeImporter.OBJ

allows user to import data from more file types.

2.6. View API additions

2.6.1. Duplicating Sheets

The new enum:

- SheetDuplicateOption

allows you to indicate what information should be copied when duplicating a sheet. Its values are:

- DuplicateEmptySheet - Only copies the title block.
- DuplicateSheetWithDetailing - Copies the title block and details.
- DuplicateSheetWithViewsOnly - Copies the title block, details, viewports and contained views. The newly created sheet will reference the newly duplicated views.
- DuplicateSheetWithViewsAndDetailing - Copies the title block, details, and viewports. Duplicates the sheet's contained views with detailing. The newly created sheet will reference the newly duplicated views.
- DuplicateSheetWithViewsAsDependent - Copies the title block, details, and viewports. Duplicates the sheet's contained views as dependent. The newly created sheet will reference the newly duplicated dependent views.

The new methods:

- ViewSheet.Duplicate(SheetDuplicateOption)

- ViewSheet.CanBeDuplicated(SheetDuplicateOption)

allows you to duplicate sheets and identify sheets which can be duplicated.

2.6.2. Transforming from Model Space to View Projection Space

New methods in View and TransformWithBoundary allow you to transform between model space and a view's projection space:

- TransformWithBoundary.GetModelToProjectionTransform()
- TransformWithBoundary.GetBoundary() - Returns the boundary for the model space to view projection space transform.
- View.GetModelToProjectionTransforms() - Gets the transforms from the model space to the view projection space. Views with split crop regions have more than one transform.
- View.HasViewTransforms() - Returns true if the view reports model space to view projection space transforms. Schedules and legends, for example, do not report any.

2.6.3. Transforming from View Projection Space to Sheet Space

Two new methods allow you to transform between a view's projection space and sheet space:

- Viewport.GetProjectionToSheetTransform()
- Viewport.HasViewportTransforms()

2.6.4. View Placement on Sheet

The new enum:

- ViewPlacementOnSheetStatus

indicates whether the View is placed on a Sheet. Some Views can be placed on one or more Sheets completely or partially. For example, a Schedule divided in segments, and only some of them are placed on Sheets.

The new method

- GetPlacementOnSheetStatus()

determines if this view placed on a sheet completely or partially.

2.6.5. Swapping viewports on sheets to another view

Viewports placed on sheets can now have the associated view swapped to another view in the model. The property:

- Viewport.ViewId

is now editable.

The new members:

- Viewport.ViewportPositioning - Specifies how the viewport will be positioned on the sheet when swapped to another view. Default is set to ViewportPositioning.ViewportCenter. The other option, ViewportPositioning.ViewOrigin, will set the viewport location based on the view origin.
- Viewport.IsViewIdValidForViewport() - Verifies that the Viewport can change its view id to the input viewId. True if the viewId is valid for the viewport, false otherwise.

2.7. Electrical API additions

2.7.1. Load classification API

The new read only property:

- ElectricalLoadClassification.Spare

indicates if this load classification is to be used for spare.

2.7.2. Panel schedule API

The new method:

- PanelScheduleView.SetLockSlot()

allows an application to set the lock state for a circuit slot at specific cell.

2.7.3. Electrical analytical node API

The new class:

- ElectricalAnalyticalNode

represents an electrical analytical node.

The new enum:

- ElectricalAnalyticalNodeType.NodeType - Represents the type of electrical analytical node.

The new properties:

- ElectricalAnalyticalNode.TotalLoad - Represents the total connected load.

The new methods:

- ElectricalAnalyticalNode.CanConnectToUpstream() - Verifies that the current node can connect to the upstream node.
- ElectricalAnalyticalNode.CanDisconnectFromUpstreamNode() - Verifies that the current node can disconnect from the upstream node.
- ElectricalAnalyticalNode.ClearConnections()
- ElectricalAnalyticalNode.ConnectToUpstreamNode()
- ElectricalAnalyticalNode.Create() - Creates an electrical analytical node.
- ElectricalAnalyticalNode.DisconnectFromUpstreamNode()

- ElectricalAnalyticalNode.GetAnalyticalPropertyData()
- ElectricalAnalyticalNode.GetDownstreamNodeIds()
- ElectricalAnalyticalNode.GetUpstreamNodeIds()

2.7.4. Bus data API

The new class:

- AnalyticalBusData

represents the data and parameters of analytical bus node.

The new properties:

- AnalyticalBusData.Rating - Represents the rating value of the analytical bus.
- AnalyticalBusData.Voltage - Represents the voltage value of the analytical bus.

The new method:

- AnalyticalBusData.GetTotalCurrent() - Gets the total connected current.

2.7.5. Distribution node property data API

The new class:

- AnalyticalDistributionNodePropertyData

represents the data and parameters of electrical analytical node.

The new property:

- AnalyticalDistributionNodePropertyData.NumberOfPoles

2.7.6. Equipment load data API

The new class:

- AnalyticalEquipmentLoadData

represents the data and parameters of point load node.

The new properties:

- AnalyticalEquipmentLoadData.ApparentLoad- Represents the electrical apparent load of analytical equipment load.
- AnalyticalEquipmentLoadData.PowerFactor - Represents the power factor of analytical equipment load.
- AnalyticalEquipmentLoadDataTrueLoad - Represents the electrical true load of analytical equipment load.

- AnalyticalEquipmentLoadData.LoadClassification - Represents the load classification of analytical equipment load.
- AnalyticalEquipmentLoadData.LoadType - Represents the load type of analytical equipment load.

The new enum:

- ElectricalLoadType - Represents the electrical load type.
- ElectricalAnalyticalNodeType - Represents the type of electrical analytical node.

2.7.7. Area based load type API

The new class:

- AreaBasedLoadType

represents an area based load type in Autodesk Revit.

The new properties：

- AreaBasedLoadType.ApparentPowerDensity - Represents apparent power density of area based load type.
- AreaBasedLoadType.LoadClassification - Represents load classification of area based load type.
- AreaBasedLoadType.LoadDensity - Represents the load density of area based load type.
- AreaBasedLoadType.PowerFactor - Represents the power factor of area based load type.

The new method：

- AreaBasedLoadType.Create() - creates an area based load type.

2.7.8. Area based load data API

The new class:

- AreaBasedLoadData

represents the electrical area based load data.

The new methods:

- AreaBasedLoadData.AddElectricalLoadArea() - Adds electrical load area into the area based load.
- AreaBasedLoadData.GetElectricalLoadAreas() - Gets electrical load areas which the area based load includes.
- AreaBasedLoadData.RemoveElectricalLoadArea() - Removes electrical load area from the area based load.
- Autodesk.Revit.DB.CurveElement.CreateAreaBasedLoadBoundaryLine() - Creates an area based load boundary line.

The new properties:

- AreaBasedLoadData.ApparentLoad - The electrical apparent load of the area based load.
- AreaBasedLoadData.ApparentPowerDensity - The apparent power density of the area based load.
- AreaBasedLoadData.Current - The current of the area based load .
- AreaBasedLoadData.AreaBasedLoadType - The electrical area based load type of the area based load.
- AreaBasedLoadData.LoadClassification - The load classification of the area based load.
- AreaBasedLoadData.LoadDensity - The load density of the area based load.
- AreaBasedLoadData.LoadType - The load type of the area based load.
- AreaBasedLoadData.PhasesNumber - The Phases Number of the area based load.
- AreaBasedLoadData.PowerFactor - The power factor of the area based load.
- AreaBasedLoadData.TrueLoad - The electrical true load of the area based load.
- AreaBasedLoadData.Voltage - The voltage of the area based load.

2.7.9. Electrical load area data API

The new class

- ElectricalLoadAreaData

represents the electrical load area data.

The new methods:

- ElectricalLoadAreaData.CreateElectricalLoadAreas() - Creates electrical load areas on all the empty plan circuits of the given level.
- ElectricalLoadAreaData.HasCircuitsWithoutElectricalLoadAreas() - Checks whether there are any empty plan circuits in which there are no electrical load areas.
- ElectricalLoadAreaData.GetAreaBasedLoadIds() - Gets the area based load ids of the electrical load area to be included.

2.7.10. Area based load boundary line data API

The new class

- AreaBasedLoadBoundaryLineData

wrapper class used to access area based load boundary line related data.

The new properties:

- AreaBasedLoadBoundaryLineData.TopLevelId - The top level id of the area based load boundary line.
- AreaBasedLoadBoundaryLineData.BottomLevelId - The bottom level id of the area based load boundary line.

The new methods:

- AreaBasedLoadBoundaryLineData.GetLevelIdsInRange() - Returns level ids between the top level and the bottom level (including the top level and the bottom level) of the area based load boundary line.

- AreaBasedLoadBoundaryLineData.IsElevationWithinRange() - Checks whether the given elevation is between the bottom level and the top level(including the bottom level and the top level) of the area based load boundary line.
- AreaBasedLoadBoundaryLineData.IsLevelWithinRange() - Checks whether the given level is between the bottom level and the top level (including the bottom level and the top level) of the area based load boundary line.

2.7.11. Analytical transfer switch data API

The new class:

- AnalyticalTransferSwitchData

represents the data and parameters of electrical analytical transfer switch.

The new properties:

- AnalyticalTransferSwitchData.CurrentRating - The current rating value of the electrical analytical transfer switch.
- AnalyticalTransferSwitchData.Voltage - The voltage value of the electrical analytical transfer switch.

The new method:

- AnalyticalTransferSwitchData.GetTotalCurrent() - Gets total connected current of the electrical analytical transfer switch.

2.7.12. Analytical Power Source API

The new class:

- Autodesk.Revit.DB.Electrical.AnalyticalPowerSourceData

represents the data and parameters of an analytical power source node.

It has the following properties:

- Autodesk.Revit.DB.Electrical.AnalyticalPowerSourceData.TotalConnectedCurrent - The total connected current of the analytical power source.
- Autodesk.Revit.DB.Electrical.AnalyticalPowerSourceData.Voltage - The voltage value of the analytical power source.

2.8. Mechanical API additions

The new class:

- Autodesk.Revit.DB.Mechanical.ZoneElementDomainData

a base class for specific domain requirements for a zone.

The new method:

- Autodesk.Revit.DB.Mechanical.Zone.CreateAreaBasedLoad()

creates a new instance of an area based load and adds it to the document.

2.9. ElementId API additions

2.9.1. Converting strings to ElementId

The new methods:

- ElementId.Parse() - Parse the string representation of the id into a corresponding ElementId. If the string represents Autodesk.Revit.DB.ElementId.InvalidElementId it will be returned. Autodesk.Revit.Exceptions.InvalidOperationException is thrown when the string cannot be parsed into an ElementId.
- ElementId.TryParse() - Same as Parse but returns false on failure rather than throwing an exception. If the parse fails, the ElementId returned is undefined.

2.10. Element API additions

2.10.1. IsModifiable property

The new property:

- Element.IsModifiable

identifies whether an element can be modified or not. The state of the property depends on the document state. For example, active edit modes can make IsModifiable false for many elements.

2.10.2. Elements with multiple ExternalResourceReferences

Material assets and Decal images are supported with external resource references now.

The new methods:

- GetExternalResourceReferencesExpanded() - Gets the expanded map of the external resource references referenced by the element.
- GetExternalResourceReferenceExpanded() - Gets the collection of ExternalResourceReference associated with a specified external resource type.

allow users to access the ExternalResourceReferences for an element which might contain multiple references for a specific type. This allows working with Material assets and Decal images, for example.

2.10.3. Category

The new property:

- Autodesk.Revit.DB.BuiltInCategory.Category

the Autodesk.Revit.DB.BuiltInCategory value for the category or Autodesk.Revit.DB.BuiltInCategory.INVALID if the category is not a built-in category.

2.11. Document API additions

2.11.1. Elements changed since a previous version

The new class:

- Autodesk.Revit.DB.ChangedElements

allows users to see which elements have changed since a previous version of the document. Note that the version refers to a DocumentVersion object and not the Revit release which the document was last saved in.

The new method:

- Document.GetDocumentVersion()

gets the current document version.

The new method:

- Document.GetChangedElements()

returns a collection of the elements which have changed between the input version and the document's current version.

2.12. Annotation API additions

2.12.1. Tag Leader API

The new enumeration:

- LeadersPresentationMode

allows users to set how leaders should be displayed on a tag. It has the following values:

- ShowAll
- HideAll
- ShowOnlyOne
- ShowSpecificLeaders

The new property

- IndependentTag.MergeElbows

identifies if the leader's elbows are merged or not

The new methods

- IndependentTag.IsLeaderVisible()

returns whether the leader that points to the specified reference is visible or not.

- IndependentTag.SetIsLeaderVisible()

sets the visibility of the leader that points to the specified reference to be visible or not.

2.13. Energy Analysis API additions

The new property:

- EnergyAnalysisSpace.Volume

this value is the enclosed volume measured by interior bounding surfaces if the spaces are created with the mode 'Use Rooms or Spaces'. Otherwise, this value is the average of the analytical volume and the voxel volume. Note that the analytical volume is measured by the center plane of walls and the top plane of roofs and floors, and the voxel volume is measured by the number of enclosed unit cubes.

The new method

- EnergyAnalysisSurface.GetPolyloops()

gets the collection of planar polygons describing the surface/opening geometry. A collection of polyloops (planar polygons) describing the opening geometry as described in gbXML. The geometry is currently measured per analytical(center-line).

2.14. Slab API additions

The new method:

- SlabShapeEditor.CreateCreasesFromFoldingLines()

allows users to pick edges on a floor and convert them from folding lines to split lines.

2.15. Applications API additions

2.15.1. Application.ShowGraphicalOpenEndsAreaBasedLoadBoundary Disconnects

The new property

- Application.ShowGraphicalOpenEndsAreaBasedLoadBoundaryDisconnects

indicates whether or not to show the graphical open ends for Area Based Load Boundary disconnects.

2.16. Sketched Elements API additions

2.16.1. Editing Sketches with SketchEditScope

The new method

- SketchEditScope.StartWithNewSketch()

starts a sketch edit mode for an element which, at this moment, doesn't have a sketch. Some surface Revit elements (like some Walls or some Analytical Elements) does not have a valid sketch all the time so in order to edit them, we have to create a valid sketch first.

2.16.2. Dimension creation in Sketch Edit mode

The method:

- Document.NewDimension()

and its overrides can now support creation of dimensions in Sketch Edit mode. If users need to create dimensions in the sketch, they must call this method under sketch edit mode, passing the view parameter as null and using sketch entities as dimension references. The method signatures were not changed but descriptions have been updated.

The method:

- Document.NewFamilyInstance()

and its overrides can now support creation of dimensions in Sketch Edit mode. If users need to create dimensions in the sketch, they must call this method under sketch edit mode, passing the view parameter as null and using sketch entities as dimension references. The method signatures were not changed but descriptions have been updated.

2.16.3. Filled Region for sketch plane

The new method:

- FilledRegion.Create() - Creates a filled region on a sketch plane in a 3d family.

2.16.4. Boundary Validation for sketched elements

The new methods:

- BoundaryValidation.IsValidBoundaryOnView() - Checks that a curve loop boundary is valid on the view's sketch plane.
- BoundaryValidation.IsValidBoundaryOnSketchPlane() - Checks that a curve loop boundary is valid on a sketch plane.

2.17. Project Browser API additions

2.17.1. BrowserOrganization

The new property

- Autodesk.Revit.DB.BrowserOrganization.Type

represents the browser organization type.

2.17.2. ProjectBrowserOptions

The new class:

- ProjectBrowserOptions

provides access to settings that control Revit's Project Browser appearance and functionality. These settings are stored in the Revit.ini file. All documents in Revit instances which use this Revit.ini will follow these settings.

The new property:

- ProjectBrowserOptions.ShowViewPlacementOnSheetStatusIcons

show the icons indicating view placement on sheet status. If true, the icons indicating view placement on sheet status will be shown at every view or schedule node in the Project Browser. If false, the icons indicating view placement on sheet status will not be shown.

2.18. Structure API additions

2.18.1. Line Load

The new methods:

- Autodesk.Revit.DB.Structure.LineLoad.Create(Document aDoc, ElementId hostElemId, XYZ forceVector1, XYZ momentVector1, LineLoadType symbol) - Creates a new hosted line load within the project.
- Autodesk.Revit.DB.Structure.LineLoad.Create(Document aDoc, ElementId hostElemId, int curveIndex, XYZ forceVector1, XYZ momentVector1, LineLoadType symbol) - Creates a new hosted line load within the project.
- Autodesk.Revit.DB.Structure.AreaLoad.isValidHostId() - Indicates if the provided host id can host line loads.

2.18.2. Area Load

The new methods:

- Autodesk.Revit.DB.Structure.AreaLoad.Create() - Creates a new hosted area load within the project.

- Autodesk.Revit.DB.Structure.AreaLoad.isValidHostId() - Indicates if the provided host id can host area loads.

2.18.3. Point Load

The new methods:

- Autodesk.Revit.DB.Structure.PointLoad.Create() - Creates a new hosted point load within the project.
- Autodesk.Revit.DB.Structure.PointLoad.isValidHostId() - Indicates if the provided host id can host point loads.

2.18.4. RebarPropagation

The new class:

- RebarPropagation

a utility class containing functions that can be used to propagate rebar elements.

The new methods:

- RebarPropagation.AlignByHost() - It will copy the source rebars, will align them in the same way as how the source host is aligned to destination host and will adapt them to the destination host.
- RebarPropagation.AlignByFace() - It will copy the source rebars, will align them to the destination face based on the source face and adapt them to destination host.

2.18.5. RebarHostData

The new method:

- RebarHostData.IsReferenceContainedByAValidHost() - Identifies whether an element that contains the given reference can host reinforcement.

2.18.6. RebarCoupler

The new property:

- RebarCoupler.RotationAngle - Identifies the rotation angle of the coupler around its axis.

2.19. Selection API additions

2.19.1. Selection

The new methods:

- Autodesk.Revit.UI.Selection.SetReferences() - Selects the references. The references can be an element or a sub element in the host or a linked document.
- Autodesk.Revit.UI.Selection.GetReferences() - Returns the references that are currently selected.

2.19.2. UIApplication

The new events:

- Autodesk.Revit.UI.UIApplication.SelectionChanged - Subscribe to the SelectionChanged event to be notified after the selection was changed.
- Autodesk.Revit.UI.UIControlledApplication.SelectionChanged

2.19.3. Events

The new class:

- Autodesk.Revit.UI.Events.SelectionChangedEventArgs

the event arguments used by the SelectionChanged event.

The new methods:

- SelectionChangedEventArgs.GetReferences()- Returns the selected references.
- SelectionChangedEventArgs.GetDocument() - Returns the document associated with this event.

2.20. MEP API additions

2.20.1. ConnectorElement

The new method:

- ConnectorElement.ChangeHostReference()

changes the connector host reference by plane reference alone, or plane reference plus an edge. The plane reference alone would allow the connector position to move along the plane later, while the plane reference plus an edge fixes the connector position by the edge loop.

2.20.2. SpatialElement

The new class

- Autodesk.Revit.DB.SpatialElementDomainData

a base class for specific domain requirements for a spatial element.

The new method:

- Autodesk.Revit.DB.SpatialElement.GetSpatialElementDomainData()

gets the domain data for the spatial element. The domain data contains information of different spatial elements, such as electrical load area. Currently room/space/area don't have domain data.

2.20.3. Flipping fabrication parts

The new method:

- Autodesk.Revit.DB.FabricationPart.Flip()

flips a directionally oriented fabrication part (tees, crosses, valves, dampers, etc.) to the opposite direction. Existing connections will be maintained. Disconnect warnings will be posted if the connection cannot be maintained.

2.21. Parameter API additions

2.21.1. ParameterUtils

The new method:

- Parameterutils.IsBuiltInParameter()

checks whether an ElementId identifies a built-in parameter. An ElementId identifies a built-in parameter if it corresponds to a valid BuiltInParameter value.

2.22. Print API additions

2.22.1. IViewSheetSet

Autodesk.Revit.DB.IViewSheetSet now supports ordered view sheet list inside. You can now access or arrange the list utilizing the new APIs. Lists can be ordered either automatically or manually, which is controlled by the property IViewSheetSet.IsAutomatic. If the list in under automatic mode, the list will be ordered by Sheet/View organization. You can set them with IViewSheetSet.SheetOrganizationId and IViewSheetSet.ViewOrganizationId. For Manual ordering, you can set IViewSheetSet.OrderedViewList.

- IViewSheetSet.OrderedViewList – Views and sheets ids in order. The order is affected by Sheet/ViewOrganizationId and IsAutomatic. Throws Autodesk.Revit.Exceptions.ArgumentNullException when the input ordered view list is null.
- IViewSheetSet.SheetOrganizationId – Autodesk.Revit.DB.ElementId to the Autodesk.Revit.DB.BrowserOrganzation for sheets. If IViewSheetSet.IsAutomatic is true, Sheets will be ordered by SheetOrganization.
- IViewSheetSet.ViewOrganizationId – Autodesk.Revit.DB.ElementId to the Autodesk.Revit.DB.BrowserOrganzation for non-sheet views. If IViewSheetSet.IsAutomatic is true, Views will be ordered by ViewOrganization.
- IViewSheetSet.IsAutomatic: bool – Represents automatic or manual ordering of the list.