What’s New

RobotStudio SDK

5.15.01

Revision: -

1 What's New in 5.15.01?

Overview

This section contains information on the new features of RobotStudio SDK 5.15.01.

1.1 Project templates for Visual Studio 2012

RobotStudio SDK now contains Visual Studio 2012 project templates for Add-Ins and SmartComponent projects.

1.2 Application Id

A new optional element `<ApplicationId></ApplicationId>` has been introduced in the .rsaddin file.

This is useful if multiple versions of the same Add-In is installed on the same machine. If the application id is not specified, all versions of the Add-In will be loaded. When the application id is specified, RobotStudio will recognize that it has found several instances of the same Add-In, and will only load the first one, and inform the user that more Add-Ins with the same application id was found.

It is recommened that all Add-Ins specify a unique application id of the format "com.organization.department.product".

It is also recommended to prefix your ProjectObject attribute keys with your application name in order to avoid collisions with attributes managed by other Add-Ins.

1.3 Loading Add-Ins downloaded from the Web

For security reasons the .NET 4.0 runtime prevents loading of assemblies downloaded from the Web under ceratin circumstances, which results in an exception being thrown.

This exception was not handled properly by RobotStudio, and may have caused problems for users downloading Add-Ins from RobotApps for example.

RobotStudio now gives you the ability to either unblock the assembly if you trust it, and continue load it, or skip loading it, if you do not trust it.

This article on MSDN contains more information about the security model change between .NET 3.5 and 4.0.

1.4 SmartComponents installed under Program Files are now trusted

A SmartComponent with Code-Behind, installed under *Program File/Program Files x86*, is now trusted by RobotStudio. This means that when loading a non signed SmartComponent from such trusted location, a warning dialog will not be displayed.

If you create an installer for your application and the users installs it, she trusts your entire application, including the Code-Behind, hence it is not necessary to sign it.

1.5 Improved validation of .rsaddin file validation

A message will be displayed in the output window if RobotStudio fails to load a PowerPac because the correct entrypoint is not specified in its .rsaddin file.

1.6 Launch RAPID Editor with module from controller memory

The method `ControllerManager.ShowControllerUserInterface(string url)` has been improved to accept paths to RAPID modules and Configuration types in the controller memory.

For example you can launch an instance of the RAPID Editor, displaying a RAPID module in a connected real or virtual controller;

```
ControllerManager.ShowControllerUserInterface("{0CE62073-0B82-4232-
B048-D93FED20A606}/RAPID/T_ROB1/Module_1")
```

To open a module on the PCs file system;

```
ControllerManager.ShowControllerUserInterface("C:\\Users\\user\\Module
1.mod")
```

Note: It is also possible to specify the path to a configuration domain, which will open the Configuration Editor instead.

1.7 All new types and methods

Overview

This section contains information about all the new types in the API, and existing types which has been extended with new methods. Each updated namespace has a separate chapter. For extended types, the name of the type is in bold face, followed by its new methods.

1.7.1 ABB.Robotics.RobotStudio.Environment

Extended types

```
class ExecuteCommandEventArgs
public CancellationToken CancellationToken
```

1.7.2 ABB.Robotics.RobotStudio

Extended types

```
class DataRecorderBase
void Start()
void Stop()

class ProjectSelection
void AddRange(IEnumerable<Object> selections)
```

1.7.3 ABB.Robotics.RobotStudio.Controllers

New types

```
class BuiltInOnlineControllerSourceSignals
class DataRecorderOnlineControllerSource
class OnlineDataRecorder
enum BuiltInOnlineDataRecorderMotionSignal
```

Extended types

```
class ControllerManager
BuiltInDataRecorderSignals
BuiltInDataRecorderSignals
OnlineDataRecorder DataRecorder
```

1.7.4 ABB.Robotics.RobotStudio.Stations

New types

```
enum MappingFailure
```

Extended types

```
class GraphicControl
void Clear(Boolean disposeChildren)
void Remove(GraphicComponent graphicComponent,
Boolean dispose)

class RsIrc5Controller
MappingResult
AssociateMechanismWithMechanicalUnits(Mechanism
mechanism, RsMechanicalUnit[] mechanicalUnits,
Boolean updateTransform)
MappingResult
ChangeMechanismInMechanicalUnits(Mechanism
mechanism, RsMechanicalUnit[] mechanicalUnits)
```

1.7.5 ABB.Robotics.RobotStudio.Forms

Extended types

```
DirectionControl
RefCoordSys RefCoordSys
class GraphicControl
Bitmap ScreenShot(ProjectObject obj, Int32
width, Int32 height, ScreenshotOptions
options)
class GraphicPicker
Boolean MultiJogEnabled
```

2 What’s New in 5.15.00.01?

Overview

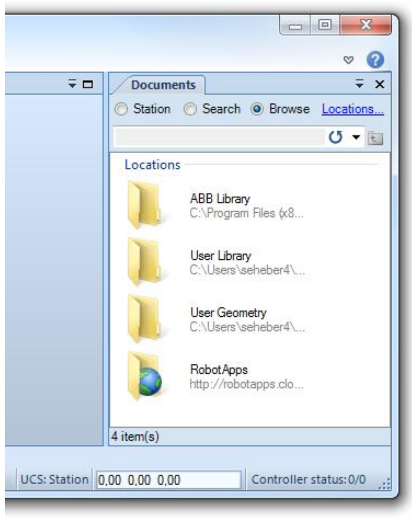
There are no new features in 5.15.00.01.

3 What's New in 5.15?

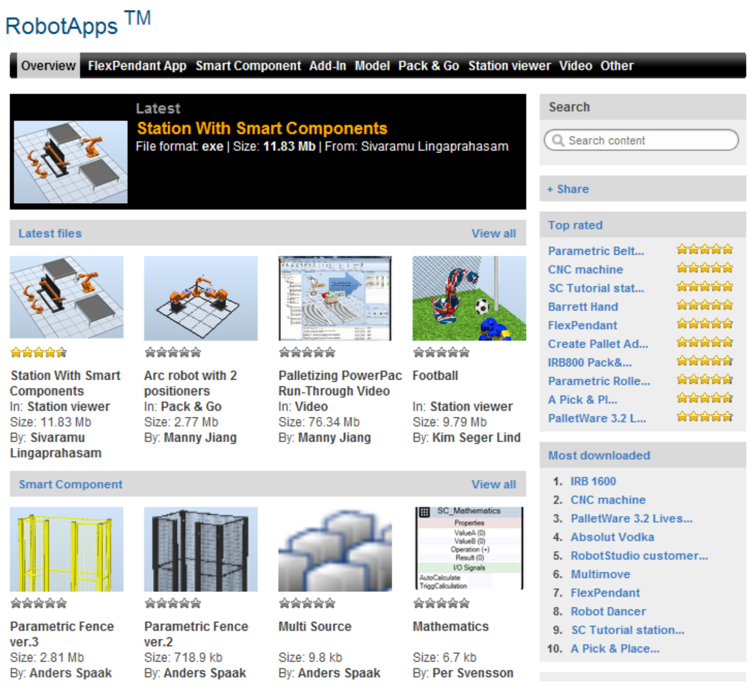
Overview

This section contains information about new features and corrected problems in 5.15. A selected set of the most important new APIs are explained. For a complete list of all new types and extended types, see 3.6.

3.1 RobotApps™ integrated in Document Window



Library files, including SmartComponents, and CAD models published on RobotApps™, is now accessible directly from the `DocumentManager` in the API and the Document Window in the RobotStudio user interface. You can publish content to the RobotApps web site under http://www.abb.com/roboticssoftware


RobotApps ™

Overview | FlexPendant App | Smart Component | Add-In | Model | Pack & Go | Station viewer | Video | Other

Latest
Station With Smart Components
File format: exe | Size: 11.83 Mb | From: Sivaramu Lingaprahasam

Latest files — View all

Station With Smart Components
In: Station viewer
Size: 11.83 Mb
By: Sivaramu Lingaprahasam

Arc robot with 2 positioners
In: Pack & Go
Size: 2.77 Mb
By: Manny Jiang

Palletizing PowerPac Run-Through Video
In: Video
Size: 76.34 Mb
By: Manny Jiang

Football
In: Station viewer
Size: 9.79 Mb
By: Kim Seger Lind

Smart Component — View all

Parametric Fence ver.3
Size: 2.81 Mb
By: Anders Spaak

Parametric Fence ver.2
Size: 718.9 kb
By: Anders Spaak

Multi Source
Size: 9.8 kb
By: Anders Spaak

Mathematics
Size: 6.7 kb
By: Per Svensson

Search
Search content

+ Share

Top rated
Parametric Belt...
CNC machine
SC Tutorial stat...
Barrett Hand
FlexPendant
Create Pallet Ad...
IRB800 Pack&...
Parametric Rolle...
A Pick & Pl...
PalletWare 3.2 L...

Most downloaded
1. IRB 1600
2. CNC machine
3. PalletWare 3.2 Lives...
4. Absolut Vodka
5. RobotStudio customer...
6. Multimove
7. FlexPendant
8. Robot Dancer
9. SC Tutorial station...
10. A Pick & Place...

3.2 RobotStudio 64-bit edition

RobotStudio is now available in a 64-bit version, and you can develop applications that runs in the 64-bit version.

3.2.1 Add-In target platform support

Add-ins built with the Visual Studio project setting *Platform Target* set to *Any CPU*, can be loaded in both the 32-bit and 64-bit edition of RobotStudio.

If the add-in has references to other assemblies they must be built for *Any CPU* as well.

Note: A native 64-bit process may not load native 32-bit dll's. As result hereof, an add-in referencing any native 32-bit assembly cannot be loaded in the 64-bit edition of RobotStudio.

It is the responsibility of the developer to verify that add-ins are compatible with the 64-bit edition of RobotStudio, and by default add-ins located outside the 64-bit bin folder, will not be loaded;

```
C:\Program Files (x86)\ABB Industrial IT\Robotics IT\RobotStudio
5.15\Bin64\Addins
```

You can specify which platform your add-in supports, by using the new `<Platform/>` element in your `.rsaddin` file.

For example, to specify that an add-in may only be loaded in the 32-bit edition;

`<Platform>x86</Platform>`.

Value	**Description**
x86	The add-in and its dependencies may only be loaded in the 32-bit edition.
x64	The add-in and its dependencies may only be loaded in the 64-bit edition.
Any	The add-in and its dependencies may be loaded in both the 32-bit and the 64-bit edition.

If the `<Platform>` element is not present, the default value is `x86`.

3.2.2 Add-In search paths

Add-ins is loaded from the directories, and in the order, as specified in the table below.

32-bit edition	
1	RobotStudio\Bin\Addins
2	Program Files (x86)\Common Files\ABB Industrial IT\Robotics IT\RobotStudio\Addins

64-bit edition	
1	RobotStudio\Bin64\Addins
2	RobotStudio\Bin\Addins *****
3	Program Files (x86)\Common Files\ABB Industrial IT\Robotics IT\RobotStudio\Addins *****

***** Remark: The 64-bit edition of RobotStudio will assume that add-ins outside the Bin64\Addins folder are 32-bit, and will hence not be loaded, unless otherwise stated by the `<Platform>` element in an `.rsaddin` file.

As before, an add-in can also be placed in a subdirectory with the same name as the add-in.

3.3 Support for asynchronous programming

The RobotStudio API now supports asynchronous programming using the `await` keyword for some methods.

These methods may have both a synchronous and an asynchronous variant. In this case the asynchronous variant has the suffix "Async".

For example the method `Task RsIrc5Controller.StartAsync()` returns immediately to the caller and returns a `Task` object.

Using the new `await` keyword in .NET 4.5 you can call several asynchronous methods after each other and even have branches between them in your code, without having to take care of waiting for the completion of each asynchronous operation.

Refer to MSDN for more information on `Task` and `await`.

Note: The `await` keyword was introduced in .NET 4.5, and the `Task` class in .NET 4.0.

3.3.1 Fire and forget

The simplest scenario is when you are calling an asynchronous method and is not interested in the return value – fire and forget.

```
void StartMyTwoControllers()
{
   ctrl1.StartAsync(…)
   ctrl2.StartAsync(…)
}
```

3.3.2 Waiting for the return value

If your code needs to know the return value from an asynchronous method call, in order to know what do to next you can use the `await` keyword.

By using `await` you can write asynchronous code that has a sequential flow and looks pretty much as it if is calling plain synchronous methods.

A simple use case is illustrated by the following code snippet;

```
DocumentInfo docInfo = null;
docInfo = await DocumentInfo.FromFileAsync(fileName);
// This line will not be executed until docInfo
// has been returned.
String author = docInfo.Author;
```

In the below scenario we are implementing a list of recently used station files where a thumbnail shall be displayed using the DocumentInfo class.

The code is called from a user interface event handler. In order to get a responsive user interface it is important to return as fast as possible from the event handler.

If the document info for our file is not already cached it has to be loaded from the station or file on disk, which can take any amount of time.

Maybe the file is not a network share. It may end up with a failure after 10 seconds, and that is a very long time for an event handler.

We have a command list where the thumbnails shall be displayed.

```
BackstageCommandList _recentListBox;
```

The track event is subscribed to.

```
_recentListBox.TrackedItemChanged += new
EventHandler(_recentListBox_TrackedItemChanged);
```

In the event handler `FromFileAsync` is called. Code which is not relevant for the example has been omitted.

```
async void _recentListBox_TrackedItemChanged(object sender, EventArgs e)
{
        …
        …
        if (fileName != null && File.Exists(fileName))
        {
                DocumentInfo docInfo = null;
                if (!_cachedDocInfo.TryGetValue(fileName, out docInfo))
                {
                        docInfo = await DocumentInfo.FromFileAsync(fileName);
                        _cachedDocInfo[fileName] = docInfo;
                        …
                        …
        }
}
```

The return value is stored in a variable and can be used on the next line, without having to take care of waiting for the `FromFileAsync` to return.

3.4 Online and Offline tabs merged

In RobotStudio 5.15 the Offline and Online tabs have been merged into the Controller tab.

This gives a unified user experience and a tab which is controller centric and allows the user to access all kinds of controllers whether they are part of a station or found on the network.

This opens up new possibilities for developers to create applications where the user can work with virtual as well as real controllers in a seamless way.

Note 1: If your existing add-in contains code that relies on the existence of the Online or Offline tab it must be modified.

Note 2: Each ribbon tab has a string identifier which may be used to programmatically access the tab. The identifiers are not published in the API documentation and ABB does not promise that they are constant between product releases. They are used at your own risk.

3.4.1 Station controller, virtual controllers and real controllers

The class `ControllerManager` is responsible for keeping track of and provides information about referenced real controllers on the network, referenced virtual controllers on the network, and virtual controllers being part of a RobotStudio station.

The following properties can be used to distinguish between different types of controller references.

`ControllerObjectReference.IsServicePortController`

`ControllerObjectReference.IsStationController`

`and`

`ConrollerTypeControllerObjectReference.ControllerType`

The `ControllerType` specifies three types of controllers;

`StationVC` - A virtual controller that is part of a RobotStudio station
`VC` – A virtual controller on the network
`RC` - A real controller on the network

3.4.2 Selected controller object

The currently selected controller reference can be retrieved using the `ControllerManager.SelectedControllerObject` property.

Virtual as well as real controllers can be selected.

3.4.3 Recent controllers list

The list of recent referenced controllers can be accessed using the

`ControllerManager.RecentControllers` property.

3.4.4 Adding a controller connection

Connections to controllers on the network can be added and removed using

```
Task<ControllerObjectReference>
ControllerManager.ControllerReferences.AddAsync(String url)
```

and

```
Task ControllerManager.ControllerReferences.RemoveAsync(String url)
```

3.4.5 Launching the RAPID Editor and Configuration Editor

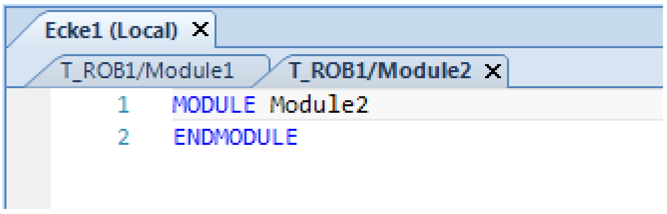
Applications can lunch the RAPID Editor or the Configuration Editor, given the URL to the RAPID module, or configuration domain.

`ControllerManager.ShowControllerUserInterface(String url)`

3.5 Two rows of document tabs

Document windows can now be organized in two levels of tabs.

All `DocumentWindow` instances with the same value of the new `Category` property will be grouped together.



```
MODULE Module2
ENDMODULE
```

Figure 1 Two document windows with the same category.

3.6 Denavit- Hartenberg parameters for mechanisms

The new method `DenavitHartenbergParameters[] GetDenavitHartenbergParameters()` can be used to retrieve DH-parameters for the specified mechanism.

DH-parameters are available for mechanisms with closed loop kinematics.

The typical use case is to retrieve DH-parameters from RobotStudio and pass them to another kinematic solver software package.

3.7 Volume intersection check on Part

With `Part.IntersectVolume (BoundingBox box, Matrix4 boxTransform)` you can check if a part intersects with a bounding box.

One use case is a SmartComponent gripper that needs to sense objects to pick, using a volume instead of a line.

3.8 All new types and methods

Overview

This section contains information about all the new types in the API, and existing types which has been extended with new methods. Each namespace has a separate chapter. For extended types, the name of the type is bold face, followed by its new methods.

3.8.1 ABB.Robotics.RobotStudio.Environment

New types

```
enum CloseButtonBehavior
interface IZoomableWindow
```

Extended types

```
class CommandBarPopup
void Show(Int32 x, Int32 y)

class CommandGroupExecuteCommandEventArgs
Task CompletionTask

class DisplayCommandGroupEventArgs
void AddSeparator()

class DocumentWindow
String Category

class ExecuteCommandEventArgs
Task CompletionTask

enum RibbonControlLayout
Hidden

StatusBarPane
AutoSizeMode AutoSizeMode
HorizontalAlignment TextAlignment

class ToolBarControl
Boolean Horizontal

class ToolWindow
CloseButtonBehavior CloseButtonBehavior

class UIEnvironment
Boolean GroupDocumentWindows
void DisableCommands(Task task)

class Window
Boolean ActiveTab

class WindowCollection
void AddDockedOrTabbed(ToolWindow window,
DockStyle dockStyle)
```

```
event
EventHandler<WindowCollectionChangedEventArgs>
Added
```

```
event
EventHandler<WindowCollectionChangedEventArgs>
Removed
```

3.8.1 ABB.Robotics.RobotStudio

New types

```
class DataRecorderSourceBaseAsync
class DataRecorderSinkBase2
interface IHasSystemId
interface IProgressCallback
```

Extended types

```
class DataRecorderSinkBase
Boolean UIVisible

class DataRecorderSourceBase2
String GetImageKey

class DocumentInfo
Task FromFileAsync(void fileName)

class Logger
void AddMessage(LogMessage msg, Boolean
bringToFront)
void AddMessage(String message, Boolean
bringToFront)

class Options
void SetFileName(String filename)
void RemoveSection(String section)

class ProjectObject
IEnumerable<ProjectObject>
FindObjects(Predicate<ProjectObject> filter,
Predicate<ProjectObject> recurse)

enum ProjectObjectChangeType
UIVisible

class RobotStudioAPI
SynchronizationContext SyncContext
```

3.8.2 ABB.Robotics.RobotStudio.Controllers

New types

`enum ControllerType`
`enum RecentControllerAvailability`
`class RecentControllerCollection`
`class RecentControllerInfo`

Extended types

`class ControllerManager`
`ControllerObjectReference SelectedControllerObject`
`RecentControllerCollection RecentControllers`
`Boolean IsStationController(String systemId)`
`Boolean ShowControllerUserInterface(String url, Object data)`
`Boolean ShowControllerUserInterface(String url)`
`class ControllerObjectReference`
`Boolean IsServicePortController`
`Boolean IsStationController`
`ControllerObjectReference Root`
`ControllerType ControllerType`
`String Name`
`String RelativeUrl`
`String SystemIdString`
`String Tag`
`String Url`
`enum ControllerObjectType`
`RAPID`
`Module`
`Routine`
`class ControllerReferenceCollection`
`ControllerObjectReference this[String systemId]`
`Task Add(void systemId)`

```
Task Add(void systemId)
Task Remove(ControllerObjectReference
controller)
```

3.8.3 ABB.Robotics.RobotStudio.Documents

Extended types

```
class DocumentManager
Task GetLocalCopyAsync(void documentInfo)
void Initialize()
```

3.8.4 ABB.Robotics.RobotStudio.Stations

New types

```
enum ControllerMappingState
struct DenavitHartenbergParameters
enum IntersectionType
enum PackAndGoFailureReason
enum PackAndGoLibraryCopyOptions
class PackAndGoResult
enum ScreenshotOptions
enum SimulationStopwatchCollection
class StationServices
struct StopwatchTrigger
enum VirtualControllerRestartMode
```

Extended types

```
class GraphicComponentCollection
void Clear(Boolean disposeChildren)
void Remove(GraphicComponent graphicComponent,
Boolean dispose)

class Mechanism
Boolean CalculateInverseKinematics(RsRobTarget
robTarget, RsWorkObject workObject, RsToolData
tool, Int32[] cfg, out Double[]
resultJointVector)
CanReach(RsRobTarget robTarget, RsWorkObject
workObject, RsToolData tool)
DenavitHartenbergParameters[]
GetDenavitHartenbergParameters()
Boolean SetJointValues(Double[] jointValues,
Boolean updateController, Boolean notify)
```

`class Part`
`IntersectionType IntersectVolume(BoundingBox box, Matrix4 boxTransform)`
`Part Load(String fileName, IProgressCallback progressCallback, Boolean surfaceModel, Boolean translateHidden, Boolean healing, DetailLevels detail)`
`class RsIrc5Controller`
`RsIrc5Controller(String systemPath)`
`ControllerMappingState MappingState`
`StartAsync(VirtualControllerRestartMode restartMode, IEnumerable<Mechanism> mechanismsToMap)`
`StartAsync(VirtualControllerRestartMode restartMode, IEnumerable<Mechanism> mechanismsToMap, Boolean checkBaseFrame)`
`event EventHandler SystemStateChanged`
`class RsIrc5ControllerCollection`
`void Add(RsIrc5Controller ctrl)`
`void Remove(RsIrc5Controller ctrl)`
`class RsJointTarget`
`void Highlight(Color color)`
`void ResetHighlight()`
`class RsMechUnit`
`Task SetBaseFrameAsync(Matrix4 baseFrame, Boolean restart)`
`event EventHandler ActivationModeChanged`
`class RsMoveInstruction`
`void Highlight(Color color)`
`void ResetHighlight()`
`class RsPathProcedure`
`Task MoveAlongAsync()`
`class RsTask`
`Boolean SetExternalAxisJointValues(Double[] jointValues, Boolean notify)`
`class RsTaskCollection`
`RsTask this[String name]`
`class SimulationConfiguration`

```
SimulationStopwatchCollection Stopwatches
```

class Simulator

```
Task StartAsync()
```

class SmartComponent

```
Boolean IsProtected
```

class Station

```
Station(Boolean setAsActive)
```

enum SystemState

```
Connecting
```

3.8.5 ABB.Robotics.RobotStudio.Forms

New types

```
class DirectionControl
```

Extended types

class NumericTextBoxArray

```
Boolean OverrideBugInSetBoundsCore
```

class GraphicControl

```
Point ProjectPointToScreen(Vector3 point)
Bitmap ScreenShot(ScreenshotOptions options)
Bitmap ScreenShot(Int32 width, Int32 height,
ScreenshotOptions options)
```