Release Notes
RobotStudio SDK
5.15.00.01

Revision: -
The information in this manual is subject to change without notice and should not be construed as a commitment by ABB. ABB assumes no responsibility for any errors that may appear in this manual.

Except as may be expressly stated anywhere in this manual, nothing herein shall be construed as any kind of guarantee or warranty by ABB for losses, damages to persons or property, fitness for a specific purpose or the like.

In no event shall ABB be liable for incidental or consequential damages arising from use of this manual and products described herein.

This manual and parts thereof must not be reproduced or copied without ABB's written permission, and contents thereof must not be imparted to a third party nor be used for any unauthorized purpose. Contravention will be prosecuted.

Additional copies of this document may be obtained from ABB at its then current charge.

© Copyright 2013 ABB All rights reserved.
ABB AB
Robotics Products
SE-721 68 Västerås
Sweden
# Release Notes

## RobotStudio SDK 5.15.00.01

### Overview

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Release Information</td>
<td>5</td>
</tr>
<tr>
<td>1.1 General</td>
<td>6</td>
</tr>
<tr>
<td>1.2 Contents</td>
<td>6</td>
</tr>
<tr>
<td>2 What's New in 5.15.00.01?</td>
<td>8</td>
</tr>
<tr>
<td>3 What's New in 5.15?</td>
<td>9</td>
</tr>
<tr>
<td>3.1 RobotApps™ integrated in Document Window</td>
<td>9</td>
</tr>
<tr>
<td>3.2 RobotStudio 64-bit edition</td>
<td>10</td>
</tr>
<tr>
<td>3.3 Support for asynchronous programming</td>
<td>11</td>
</tr>
<tr>
<td>3.3.1 Fire and forget</td>
<td>11</td>
</tr>
<tr>
<td>3.3.2 Waiting for the return value</td>
<td>11</td>
</tr>
<tr>
<td>3.4 Online and Offline tabs merged</td>
<td>12</td>
</tr>
<tr>
<td>3.5 Two rows of document tabs</td>
<td>14</td>
</tr>
<tr>
<td>3.6 Denavit- Hartenberg parameters for mechanisms</td>
<td>14</td>
</tr>
<tr>
<td>3.7 Volume intersection check on Part</td>
<td>14</td>
</tr>
<tr>
<td>3.8 All new types and methods</td>
<td>14</td>
</tr>
<tr>
<td>3.8.1 ABB.Robotics.RobotStudio.Environment</td>
<td>15</td>
</tr>
<tr>
<td>3.8.2 ABB.Robotics.RobotStudio.Controllers</td>
<td>16</td>
</tr>
<tr>
<td>3.8.3 ABB.Robotics.RobotStudio.Documents</td>
<td>18</td>
</tr>
<tr>
<td>3.8.4 ABB.Robotics.RobotStudio.Stations</td>
<td>18</td>
</tr>
<tr>
<td>3.8.5 ABB.Robotics.RobotStudio.Forms</td>
<td>20</td>
</tr>
<tr>
<td>4 Late-breaking information</td>
<td>21</td>
</tr>
<tr>
<td>4.1 Overview</td>
<td>21</td>
</tr>
<tr>
<td>5 Problems Corrected</td>
<td>22</td>
</tr>
<tr>
<td>5.1 Solved Product Defect Documents (PDD) since RobotStudio SDK 5.15</td>
<td>22</td>
</tr>
<tr>
<td>5.2 Solved Product Defect Documents (PDD) since RobotStudio SDK 5.14.03</td>
<td>22</td>
</tr>
<tr>
<td>6 Known Limitations</td>
<td>23</td>
</tr>
<tr>
<td>6.1 Development Environment</td>
<td>23</td>
</tr>
<tr>
<td>6.2 Visual Studio Tools for Applications (VSTA)</td>
<td>23</td>
</tr>
</tbody>
</table>

© Copyright 2013 ABB All rights reserved
# 7 Installation Information

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1 Hardware and Software requirements</td>
<td>24</td>
</tr>
<tr>
<td>7.2 Compatibility</td>
<td>24</td>
</tr>
</tbody>
</table>
Overview

Description

This document provides information about the new features, problems corrected, and installation of RobotStudio SDK.

Product Overview

RobotStudio SDK is included in the RobotStudio installation and can be used for free by anyone who wants to develop on the RobotStudio platform. It could be scripts, Add-Ins or applications that can provide new functionality and/or customized user interfaces in RobotStudio.

You can also combine the RobotStudio SDK with the PC SDK to communicate with real or virtual IRC5 controller functionality from an Add-In.

Custom Smart Components with Code Behind can also be developed using the RobotStudio SDK.

RobotStudio, including RobotStudio SDK, can be downloaded from the Download section.

To get started, or to learn more, visit our Developer Center: http://developercenter.robotstudio.com.

Here you will also find the User Forum, where developers discuss software problems and solutions online.
1 Release Information

1.1 General

Release Name
The release name is RobotStudio SDK 5.15.00.01 and the build number is 5.15.5156.0266.

For information about RobotStudio, please refer to the document ‘Release Notes RobotStudio 5.15.pdf’.

Release Date
The release date is 28th of January, 2013.

1.2 Contents

Overview
The installation includes software, documentation and tools as specified below.

Software
ABB.Robotics.Math.dll
ABB.Robotics.Environment.dll
ABB.Robotics.RobotStudio.dll
ABB.Robotics.RobotStudio.Controllers.dll
ABB.Robotics.RobotStudio.Documents.dll
ABB.Robotics.RobotStudio.Stations.dll
ABB.Robotics.RobotStudio.Stations.Forms.dll

Documentation

- API reference
- Sample projects
- Walkthroughs

After installation the help file can be launched from the RobotStudio Help Menu, and Windows Start Menu\Programs\ABB Industrial IT\Robotics IT\Robot Studio 5.15\SDK.

Tools

LibraryCompiler.exe – A tool for batch creation for RobotStudio libraries such as SmartComponents and Mechanisms.

Visual Studio Project Templates (C#)
RobotStudio Add-In – template for a standard Add-In
RobotStudio Smart Component – template for a Smart Component with Code Behind and XML description.
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 from the RobotApps web site under http://www.abb.com/roboticssoftware
3.2 RobotStudio 64-bit edition

RobotStudio is now available in a 64-bit version. The 64-bit version will be automatically installed on a PC that runs a 64-bit version of Windows. The 64-bit version allows large CAD models and stations viewers to be managed without hitting the memory limitation.

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>.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x86</td>
<td>The add-in and its dependencies may only be loaded in the 32-bit edition.</td>
</tr>
<tr>
<td>x64</td>
<td>The add-in and its dependencies may only be loaded in the 64-bit edition.</td>
</tr>
<tr>
<td>Any</td>
<td>The add-in and its dependencies may be loaded in both the 32-bit and the 64-bit edition.</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>32-bit edition</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RobotStudio\Bin\Addins</td>
</tr>
<tr>
<td>2</td>
<td>Program Files (x86)\Common Files\ABB Industrial IT\Robotics IT\RobotStudio\Addins</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>64-bit edition</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RobotStudio\Bin64\Addins</td>
</tr>
</tbody>
</table>
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.

```csharp
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 if it is calling plain synchronous methods.

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

```csharp
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 you’re 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
- `ControllerTypeControllerObjectReference.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 launch 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 tab rows.

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

![Two document windows with the same category.](image)

3.6 Denavit- Hartenberg parameters for mechanisms

The new method `DenavitHartenbergParameters[]` 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**

<table>
<thead>
<tr>
<th>Class/Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>class DataRecorderSourceBaseAsync</td>
<td></td>
</tr>
<tr>
<td>class DataRecorderSinkBase2</td>
<td></td>
</tr>
<tr>
<td>interface IHasSystemId</td>
<td></td>
</tr>
<tr>
<td>interface IProgressCallback</td>
<td></td>
</tr>
</tbody>
</table>

**Extended types**

<table>
<thead>
<tr>
<th>Class/Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>class DataRecorderSinkBase</td>
<td>Boolean UIVisible</td>
</tr>
<tr>
<td>class DataRecorderSourceBase2</td>
<td>String GetImageKey</td>
</tr>
<tr>
<td>class DocumentInfo</td>
<td>Task FromFileAsync(void fileName)</td>
</tr>
<tr>
<td>class Logger</td>
<td>void AddMessage(LogMessage msg, Boolean bringToFront)</td>
</tr>
<tr>
<td></td>
<td>void AddMessage(String message, Boolean bringToFront)</td>
</tr>
<tr>
<td>class Options</td>
<td>void SetFileName(String filename)</td>
</tr>
<tr>
<td></td>
<td>void RemoveSection(String section)</td>
</tr>
<tr>
<td>class ProjectObject</td>
<td>IEnumerable&lt;ProjectObject&gt; FindObjects(Predicate&lt;ProjectObject&gt; filter, Predicate&lt;ProjectObject&gt; recurse)</td>
</tr>
<tr>
<td>enum ProjectObjectChangeType</td>
<td>UIVisible</td>
</tr>
<tr>
<td>class RobotStudioAPI</td>
<td>SynchronizationContext SyncContext</td>
</tr>
</tbody>
</table>

### 3.8.2 ABB.Robotics.RobotStudio.Controllers

**New types**

<table>
<thead>
<tr>
<th>Enum/Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enum ControllerType</td>
<td></td>
</tr>
<tr>
<td>enum RecentControllerAvailability</td>
<td></td>
</tr>
<tr>
<td>class RecentControllerCollection</td>
<td></td>
</tr>
<tr>
<td>class RecentControllerInfo</td>
<td></td>
</tr>
</tbody>
</table>
## Extended types

<table>
<thead>
<tr>
<th>Class</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ControllerManager</strong></td>
<td>ControllerObjectReference, SelectedControllerObject, RecentControllerCollection, RecentControllers</td>
</tr>
<tr>
<td></td>
<td>Boolean IsStationController(String systemId)</td>
</tr>
<tr>
<td></td>
<td>Boolean ShowControllerUserInterface(String url, Object data)</td>
</tr>
<tr>
<td></td>
<td>Boolean ShowControllerUserInterface(String url)</td>
</tr>
<tr>
<td><strong>ControllerObjectReference</strong></td>
<td>Boolean IsServicePortController, IsStationController, Root, ControllerType, ControllerType, String Name, String RelativeUrl, String SystemIdString, String Tag, String Url</td>
</tr>
<tr>
<td><strong>ControllerObjectType</strong></td>
<td>RAPID, Module, Routine</td>
</tr>
<tr>
<td><strong>ControllerReferenceCollection</strong></td>
<td>ControllerObjectReference this[String systemId], Task Add(void systemId), Task Add(void systemId), Task Remove(ControllerObjectReference controller)</td>
</tr>
</tbody>
</table>
3.8.3 ABB.Robotics.RobotStudio.Documents

Extended types

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

3.8.4 ABB.Robotics.RobotStudio.Stations

New types

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

Extended types

```csharp
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()
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)
4 Late-breaking information

4.1 Overview

This section contains late-breaking information that will be included in the appropriate documents in the subsequent releases.
5 Problems Corrected

Overview
This section describes the problems solved in RobotStudio SDK.

5.1 Solved Product Defect Documents (PDD) since RobotStudio SDK 5.15
No solved PDDs.

Other defects solved

<table>
<thead>
<tr>
<th>WI ID</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>8279</td>
<td>RobotStudio SDK templates not copied to VisualStudio for 64-bit exe</td>
</tr>
<tr>
<td>8050</td>
<td>Performance to GetJointSignals is very slow</td>
</tr>
</tbody>
</table>

5.2 Solved Product Defect Documents (PDD) since RobotStudio SDK 5.14.03
No solved PDDs.
6 Known Limitations

6.1 Development Environment

The RobotStudio API is not thread safe

Access to the RobotStudio API is not inherently thread safe. Only access the API from the thread that your Add-In was called from by RobotStudio. If multiple threads manipulate the object model it can be left in an inconsistent state.

6.2 Visual Studio Tools for Applications (VSTA)

*Breakpoints are not supported in VSTA

It is not possible to debug VSTA projects using breakpoints. When selecting Start Debugging from the Debug menu of VSTA, then the added breakpoints will be disabled.

**Workaround:** Create an add-in using Visual Studio that does not use VSTA.

Properties and methods that use the type System.Drawing.Color will not work in VSTA.

This is a limitation on the Visual Studio Tools for Applications (VSTA) environment.

**Note:** The VSTA-class VSTABridge can be used to work around this problem, see API documentation.

Static events cannot be called from applications developed in VSTA.

This affects for example the Simulation – Tick event.

**Workaround:** Create a standard add-in if static events are to be used. Alternatively, use the VSTABridge class that can workaround this problem, see API documentation.

Debugging of VSTA Applications

When debugging a VSTA application that adds menu items to the RobotStudio environment, then the menu will not be removed when the program execution stops. This may cause multiple entries of the same menu to be added in RobotStudio. This only affects VSTA add-ins being debugged and not completed VSTA add-ins.

**Workaround:** Restart RobotStudio to remove the extra menus.

VSTA Library add-ins not available

In the Add-ins browser there is a folder for so-called VSTA Library add-ins. This feature is not available.

Use Visual Studio 2010 or 2012 Express for advanced add-in

The purpose of VSTA is to write custom actions and minor utilities. For advanced add-in development use Visual Studio 2010 Express that can be downloaded free of charge from [http://www.microsoft.com/express/](http://www.microsoft.com/express/)

RsLoadData does not work from VSTA

**VSTA limitation**

The ‘FindDataDeclarationsByType’ method used in VB.net throw an exception Limitation added to API doc.
7 Installation Information

Overview
In order for the Visual Studio project templates to be installed, Visual Studio 2010 needs to be installed on the computer before you install RobotStudio SDK.

To install RobotStudio SDK click RobotStudio on the RobotWare & RobotStudio DVD. If you select the default installation option Complete, RobotStudio SDK will be installed. The .NET assemblies and Visual Studio template can be optionally installed, while the documentation is always installed. If you do not want to install the assemblies and templates, select the installation option Custom and uncheck the feature RobotStudio/SDK.

7.1 Hardware and Software requirements

Software requirements
Operating system:
Same as RobotStudio – Please refer to RobotStudio Release Notes

Supported development environment
Microsoft Visual Studio 2010
The Visual Studio project templates works only for the English version of Visual Studio.

NOTE!
As RobotStudio 5.15 is built on .NET Framework 4.0, Microsoft Visual Studio 2008 can not be used for RobotStudio 5.15 application development.

It is possible, but not supported, to develop RobotStudio Add-Ins with any .NET development environment, such as Visual Studio 2010 Express or SharpDevelop, but there will be no project templates available. The RobotStudio API documentation also assumes that Visual Studio 2010 is used, which means that information about VisualStudio project settings etc has to be adopted to the other development environment.

7.2 Compatibility
The APIs in the RobotStudio SDK are backwards compatible.