

---

# New Features and Migration for Experiment and Test Software

This document provides you with a brief overview of the new features and necessary migration steps for the experiment and test software of dSPACE Release 3.5.

## **New features**

For a description of the new features in the experiment and test software of dSPACE Release 3.5, refer to *New Experiment and Test Features* on page 2.

## **Migration**

To read up on the changes you may have to perform when you migrate from previous releases to dSPACE Release 3.5, refer to *Migrating to dSPACE Release 3.5* on page 5.

## New Experiment and Test Features

dSPACE Release 3.5 has many new features and enhancements for dSPACE's experiment and test software. See:

- *ControlDesk Automation* on page 2
- *Python Core* on page 3

### ControlDesk Automation

ControlDesk for dSPACE Release 3.5 comes with the following new features and enhancements:

- The `Processorplatform` class now has a new method `SaveFlightrecorderData`.
- A new exception is raised under XP if two users want to use ControlDesk automation at the same time.
- Button instruments now have a new multiple line caption property.
- Office XP (Office 10) is supported.
- The value property of an **Invisibleswitch** instrument could now be set.
- Application class has two new properties: `FullscreenMode` and `WorkbookMode` to get and set the full screen or workbook style of the main window.
- The `MacroDelay` function is now included in the `cdautomationlib`.

## Python Core

The Python core on dSPACE Release 3.5 comes with the following new features and enhancements:

### Unicode support

- There are now two string types: string and unicode. A unicode string is only valid with a given encoding. If a unicode string is to be interpreted as a normal string, the default encoding in the dSPACE installation is the active system encoding and could be queried with:

```
import sys
sys.getdefaultencoding()
```

- Unicode strings are prefixed with a u



All automation methods (ControlDesk, Office) return strings encoded with the default encoding. This means that values retrieved from any office products are not unicode strings.

### New import mechanism

- A module can be imported with a different name, for example, `import X as Y`. Therefore, the module name could differ in comparisons.
- `from moduleXY import *` imports only the classes, variables and functions which are listed in a `__all__` variable. If no such variable is present, all elements are imported.
- Long ints and ints are merged into one type. There is no need to set an L in front of a long integer anymore. Therefore, the following code would return an empty list: `str(2L)[-1]`
- All exceptions must be derived from the standard exception. String exceptions will be removed in the next version. To define a new exception, the following code is at least required:

```
from exceptions import StandardError
class MyError(StandardError):
    pass
```

### Nested scopes

Python now has multiple nested scopes. This means that a namespace hierarchy for functions can be built up. The following code would fail in Python 1.5.2, however it works in Python 2.2.1

```
A = 0
def F():
    def F2():
        C = 2
        print A,B,C
    B = 1
    F2()
```

Therefore, `from XX import *` is only allowed in the global scope since otherwise the name look-up would fail with nested scopes.

### Augmented assignment

The following operators are now available for the basic types `+=`, `-=`, `*=`, `^=`, `|=`, `&=`

### String methods

- The functions from the string module are now attached to the string object itself. These functions still return a copy of the changed string

```
>>> a = "asas"
>>> a.replace("a", "b")
'bsbs'
```

- A new format style is available when using the `%` operator; `'%r'` inserts the `repr()` of its argument.

### New exceptions

New exceptions were defined:

- `UnboundLocalError` if a variable is used before assignment
- `TabError` if there is a mix of tab and spaces in the indentation
- `IndentationError` instead of syntax error if the indentation does not match

### Further information

Further information is available at:

- <http://www.python.org/doc/2.2.1/whatsnew/>
- <http://www.amk.ca/python/2.1/>
- <http://www.amk.ca/python/2.0/>

# Migrating to dSPACE

## Release 3.5

Note the following points when migrating from dSPACE Release 3.4 to dSPACE Release 3.5:

- *Migrating to ControlDesk Automation* on page 5
- *Migrating to Python 2.2* on page 6
- *Migrating the IOCLib* on page 7

## Migrating to ControlDesk Automation

### Obsolete functions

The following changes will affect existing user Python code. There are a few more warnings for obsolete functions in dSPACE Release 3.5. These methods will not be supported in future versions of ControlDesk. The following tables show the old method and the corresponding new method that should now be used. You should change the code as shown in the tables.

The following table shows the new obsolete methods in dSPACE Release 3.5.

Old Method	New Method
Layout.AddInstrument(...)	Layout.Instruments.Add(...)
Layout.GetInstrument(...)	Layout.Instruments.Item(...)
Layout.RemoveInstrument(...)	Layout.Instruments.Remove(...)
Layout.Size(...)	Layout.Top, Layout.Width, Layout.Height, Layout.Left
Layout.GetBaseName(...)	Layout.Name or Layout.PathName
Layout.SetProperties(...)	Layout.AmbientForeColor, Layout.AmbientBackColor, Layout.ContentsWidth, Layout.ContentsHeight

As a reminder the following table shows the obsolete methods since dSPACE Release 3.0.

Old Method	New Method
HardwareManager	PlatformManager
HardwareManager().GetBoardNames(...)	PlatformManager().Platform.Names()
HardwareManager().LoadApplication(...)	PlatformManager().Platforms.Item(Name).Load(FileName)
HardwareManager().LoadMPApplication(...)	PlatformManager().Platforms.Item(Name).Load(FileName)
HardwareManager().BatchMode(...)	PlatformManager().Mode = cdacon.pfmcBatch
HardwareManager().GuiMode(...)	PlatformManager().Mode = cdacon.pfmcGui
HardwareManager().StartApplication(...)	PlatformManager().Platforms.Item(Name)..Start()
HardwareManager().StopApplication(...)	PlatformManager().Platforms.Item(Name)..Stop()
HardwareManager().SetScoutMode(...)	PlatformManager().Mode
VariableBrowser().LoadTrcFile(...)	VariableBrowser().Load(...)
VariableBrowser().LoadSdfFile(...)	VariableBrowser().Load(...)
VariableBrowser().AssignBoard(...)	VariableBrowser().AssignPlatform()
ExperimentManager().ViewMode	ExperimentManager().Properties.ViewMode
ExperimentManager().DoConsistencyCheck	ExperimentManager().Properties.ConsistencyCheck
ExperimentManager().ShowMsgBoxForConsistencyCheck	ExperimentManager().Properties.MessageBoxForConsistencyCheck
ExperimentManager().New(...)	ExperimentManager().Experiments.New(...)
ExperimentManager().Open(...)	ExperimentManager().Experiments.Add(...)
ExperimentManager().Save(...)	ExperimentManager().ActiveExperiment.Save()
ExperimentManager().SaveAs(...)	ExperimentManager().ActiveExperiment.SaveAs()
ExperimentManager().Close(...)	ExperimentManager().Experiments.RemoveAll()
ExperimentManager().ImportFile(...)	ExperimentManager().ActiveExperiment.Files.Add(...)
ExperimentManager().RemoveFile(...)	ExperimentManager().ActiveExperiment.Files.Remove(...)
ExperimentManager().AddToExperiment(...)	ExperimentManager().ActiveExperiment.Files.Add(...)
ExperimentManager().RemoveFromExperiment(...)	ExperimentManager().ActiveExperiment.Files.Remove(...)
ExperimentManager().DeleteFromExperiment(...)	ExperimentManager().ActiveExperiment.Files.Remove(...)
ExperimentManager().Zip(...)	ExperimentManager().ActiveExperiment.Files.Zip(...)
Application().ExecuteCommand()	Not used anymore

### Removed file

The file `endprocesses.py` file is removed from the installation

## Migrating to Python 2.2

### New folder structure

The Python installation is now located in

```
PythonRoot = [DSPACE_ROOT]\Common\Python22\Core
```

### Installed versions

The following versions are installed:

- Python 2.2.1
- PythonWin build 146
- wxPython Version 2.3.2.1

- pyXML 0.8.1
- Self compiled Python DLLs**

Self compiled Python DLLs (PYD) must be compiled with the new library. The new import library is located in the [PythonRoot]\libs folder and the includes in [PythonRoot]\include.
- Tuples in function parameters**

The use of tuples in function parameters has changed. There is no intrinsic conversion anymore if multiple elements are used in list functions. See the following example:

```
Python 1.5.2: X.append("a", "b"), X.index("a", "b")
Python 2.2.1: X.append(["a", "b"]), X.index(["a", "b"])
```
- New variables**

New Boolean variables True and False are defined.

### Migrating the IOCILib

The demo for the "IOCILib" needs a newer wxPython version, which could not be included in the standard installation. The required Version 2.3.4.2 is located in %dSPACE\_ROOT%\Demos\ControlDesk\ControlDeskTestautomation\AccessingExternalDevices\iocilib\Examples.

