ANSYS ACT 19.0 Migration Notes

Copyright and Trademark Information

Disclaimer Notice

U.S. Government Rights

Third-Party Software

Table of Contents

ANSYS ACT 19.0 Migration Notes

As improvements are made to ACT APIs and the way that they display and transmit data, great efforts are taken to ensure that changes are backwards-compatible. For your convenience, this section lists API changes that might impact your existing extensions so that you can determine if any action is necessary before migrating them to ANSYS 19.0.

Changes to Mechanical's Automation API methods for user interface transactions

The new method with `Transaction()` replaces the now deprecated method `with ExtAPI.DataModel.Tree.Suspend()`. When using Mechanical's Automation API to create or modify many objects in the Mechanical tree, the method `with Transaction()` allows you to define the scope of a user interface transaction so that only one refresh is performed at the end of the transaction. By eliminating refreshes after every method call, this method significantly improves scripting performance. Sample code follows:

```
with Transaction():
    contacts = ExtAPI.DataModel.Project.Model.Connections.Children[0]
    for contact in contacts.Children:
        contact.ContactType = ContactType.NoSeparation
```

Setting phase angle tabular data for a remote force

In the Mechanical API, the exposed property `PhaseAngle` could not be used to set phase angle tabular data for a remote force. The issue was fixed. Additionally, the return type was changed from `quantity` to `field` so that phase angle tabular data can now be set successfully.

Modification to the property `ThreadAngle` for a contact region

In the Mechanical API, the return type for the property `ThreadAngle` was changed from `double` to `quantity`. With this change, the property `ThreadAngle` now behaves like other angle properties.

Semantics change for comparing Mechanical tree objects

Due to a change in the automation API, to compare Mechanical tree objects, use the method `Equals()` or `"=="` (in IronPython only) rather than using the keyword `is`.

In the sample code that follows, two variables reference the same project in the Mechanical tree:

```
project = ExtAPI.DataModel.Project
same_project = ExtAPI.DataModel.Project
```

The operation `project is same_project` will return `False`. Instead, use `project.Equals(same_project)` or `project == same_project`.

Note: ACT has superseded the ANSYS Workbench Software Development Kit (SDK) and External Connection Add-In as the best-in-class tool set for customizing ANSYS products. As of 19.0, support for the SDK and External Connection Add-in has ended. If you have used these deprecated tools for Workbench customizations, see the [ANSYS SDK and External Connection Add-in Migration Guide](#) on the [ACT Resources page](#) of the ANSYS Customer Community for migration information.