

tempt to save as HDF5.

- If there is a conflict between a wave and a variable, if overwrite is off, you get an error; if overwrite is on, the dataset for the variable overwrites the dataset for the wave so the wave is not saved to the resulting HDF5 file. In Igor Pro 9.00, when writing an HDF5 packed experiment file, Igor turned overwrite on; in Igor Pro 9.01 we changed overwrite to off so that an error will be flagged in this situation.

It is conceptually possible for Igor to handle these conflicts by changing the name of one of the conflicting objects when the file is saved, somehow marking the fact that this change was made, and reversing the process when the file is loaded. When we tried to implement this scheme, we found that it added a significant amount of complexity to already complex code. Adding complexity introduces the possibility of creating new bugs and slowing operations down. Since these name conflicts are rare, the downside of implementing such a workaround outweighed the benefits so we decided to decline to support name conflicts in HDF5 files.

Graphics Technology

As of version 7, Igor Pro is based on the cross-platform Qt framework and, by default, Igor uses Qt for graphics. However, for special purposes, Igor provides access to platform-native graphics.

You should avoid native graphics and stick with Qt graphics if possible because it is the focus for future development. Native graphics is provided mainly for emergency use.

You can select native graphics on a global basis (all windows are affected) in two ways:

1. From the Miscellaneous category of the Miscellaneous Settings dialog (Misc→Miscellaneous Settings menu item).
2. By executing

```
SetIgorOption GraphicsTechnology=n
```

where *n* is interpreted as follows:

<i>n</i> =0	Default (currently Qt)
<i>n</i> =1	Quartz on Macintosh, GDI+ on Windows
<i>n</i> =2	Not used on Macintosh, GDI on Windows
<i>n</i> =3	Qt

Unlike most other SetIgorOption cases, this change is saved to preferences on disk and applies to future Igor sessions.

In addition to changing the global graphics technology setting, you can change individual windows using

```
SetWindow winName, graphicsTech=n
```

where *n* is the same as above. *winName* can be *kwTopWin* or the actual name of a window. Currently this setting is saved for graphs only but that is subject to change.

Over time, it is our intention that Qt graphics will replace the other technologies and the SetIgorOption GraphicsTechnology option may no longer be supported.

Graphics Technology on Windows

In general, you should use the default graphics (GraphicsTechnology=0) which is currently Qt graphics. If you experience problems with graphics, you might try GraphicsTechnology=1 (GDI+) or 2 (GDI). High-resolution displays are supported using Qt graphics only.

On Windows, GraphicsTechnology=1 utilizes the GDI+ interface. This provides support for transparency and other advanced features. Unfortunately, GDI+ is very slow, especially when rendering text.