

```
Wave/Z w = jack
Differentiate fred /D=w      // Creates a wave named w in current data folder

Wave/Z w = root:FolderA:jack
Differentiate fred /D=w      // Creates a wave named w in current data folder

Wave/Z jack = root:FolderA:jack
Differentiate fred /D=jack // Creates a wave named jack in current data folder

STRUCT MyStruct s           // Contains wave ref field w
Differentiate fred /D=s.w    // Creates a wave named w in current data folder
```

In a situation like this, you should add a test using `WaveExists` to verify that the destination wave is valid and throw an error if not or otherwise handle the situation. For example:

```
Wave/Z w = root:FolderA:jack
if (!WaveExists(w))
    Abort "Destination wave does not exist"
endif
Differentiate fred /D=w
```

As noted above, when you use a simple name as a destination wave, the Igor compiler automatically creates a wave reference. If the automatically-created wave reference conflicts with a pre-existing wave reference, the compiler generates an error. For example, this function generates an "inconsistent type for wave reference error":

```
Function InconsistentTypeError()
    Wave/C w           // Explicit complex wave reference
    Differentiate fred /D=w // Implicit real wave reference
End
```

Another consideration involves loops. Suppose in a loop you have code like this:

```
SetDataFolder <something depending on loop index>
Duplicate/O srcWave, jack
```

You may think you are creating a wave named jack in each data folder but, because the contents of the automatically-created wave reference variable jack is non-null after the first iteration, you will simply be overwriting the same wave over and over. To fix this, use

```
Duplicate/O srcWave, jack
WaveClear jack
```

or

```
Duplicate/O srcWave, $"jack"/WAVE=jack
```

This creates a wave named jack in the current data folder and stores a reference to it in a wave reference variable also named jack.

### Changes in Destination Wave Behavior

Igor's handling of destination wave references was improved for Igor Pro 6.20. Previously some operations treated wave references as simple names, did not set the wave reference to refer to the destination wave on output, and exhibited other non-standard behavior.

## Programming With Trace Names

A trace is the graphical representation of a 1D wave or a subset of a multi-dimensional wave. Each trace in a given graph has a unique name within that graph.