```
   uint32 triggerDelay
   uint32 reserved[99]        // Reserved for future use
EndStructure
```

Here the triggerDelay field was added and size of the reserved field was reduced to keep the overall size of the structure the same. The AcmeDataAcqLoadPackagePrefs function would also need to be changed to set the default value of the triggerDelay field.

If you need to change the structure such that its size changes or its fields are changed in an incompatible manner then you must change your structure version, which will overwrite old preferences with new preferences.

A functioning example using this technique can be found in:

"Igor Pro Folder:Examples:Programming:Package Preferences Demo.pxp"

In the example above we store just one structure in the preference file. However LoadPackagePreferences and SavePackagePreferences allow storing any number of structures of the same or different types in the preference file. You can store either multiple instances of the same structure or multiple different structures. You must assign a unique nonnegative integer as a record ID for each structure stored and pass this record ID to LoadPackagePreferences and SavePackagePreferences. You could use this feature, for example, to store a different structure for each type of control panel that your package presents. Since all data is cached in memory you should not attempt to store hundreds or thousands of structures.

In almost all cases a particular package will need just one preference file. For the rare cases where this is inconvenient, LoadPackagePreferences and SavePackagePreferences allow each package to create any number of preference files, each with a distinct file name. All of the preference files for a particular package are stored in the same directory, the package's preferences directory. Each file can store a different set of structure. However, the code that implements this feature is not tuned to handle large numbers of files so you should not use this feature indiscriminately.

Saving Package Preferences in an Experiment File

This approach supports package preference data consisting of waves, numeric variables and string variables. It is more difficult to implement than the special-format binary file approach and is not recommended except for expert programmers and then only if the previously described approach is not suitable.

You use the SaveData operation to store your waves and variables in a packed experiment file in your package directory on disk. You can later use the LoadData operation to load the waves and variables into a new experiment.

You must create your package directory as illustrated by the SavePackagePrefs function below.

The following example functions save and load package preferences. These functions assume that the package preferences consist of all waves and variables at the top level of the package's data folder. You may need to customize these functions for your situation.

```
// SavePackagePrefs(packageName)
// Saves the top-level waves, numeric variables and string variables
// from the data folder for the named package into a file in the Igor
// preferences hierarchy on disk.
Function SavePackagePrefs(packageName)
   String packageName              // NOTE: Use a distinctive package name.

   // Get path to Packages preferences directory on disk.
   String fullPath = SpecialDirPath("Packages", 0, 0, 0)
   fullPath += packageName

   // Create a directory in the Packages directory for this package
   NewPath/O/C/Q tempPackagePrefsPath, fullPath

   fullPath += ":Preferences.pxp"

   DFREF saveDF = GetDataFolderDFR()
   SetDataFolder root:Packages:$packageName
   SaveData/O/Q fullPath           // Save the preference file
   SetDataFolder saveDF
```