

Loading and mounting a file system

We add file systems to the file server by calling the client method:

```
TInt RFs::AddFileSystem(const TDesC& aFileName) const
```

The argument `aFileName` specifies the name of the FSY component to be loaded. ESTART normally does file-system loading during file server startup.

Once it has been successfully added, a file system can be mounted on a particular drive using the method:

```
TInt RFs::MountFileSystem(const TDesC& aFileSystemName, TInt aDrive) const
```

In this method, `aFileSystemName` is the object name of the file system and `aDrive` is the drive on which it is to be mounted.

The EKA1 version of the file server requires a nominated default file system, which must be called ELOCAL.FSY. The EKA2 version of the file server places no such restriction on the naming of file systems, or in requiring a default file system.

If you are developing file systems, there are two methods available which are useful for debugging:

```
TInt RFs::ControlIo(TInt aDrive, TInt, TAny*, TAny*)
```

This is a general-purpose method that provides a mechanism for passing information to and from the file system on a specified drive. The argument `aDrive` specifies the drive number, but the assignment of the last three arguments is file system specific. Additionally, the following method can be used to request asynchronous notification of a file system specific event:

```
void RFs::DebugNotify(TInt aDrive, TInt aNotifyType,  
TRequestStatus& aStat)
```

The argument `aDrive` specifies the target drive number, `aNotify- Type`, specifies the event, and `aStat` is a reference to a request status object that is signaled when the event occurs. To trigger the notifier, the file system calls the following method, which is exported by the file server:

```
void DebugNotifySessions(TInt aFunction, TInt aDrive)
```

The argument `aFunction` specifies the event that has occurred and `aDrive` indicates the drive on which this has occurred. So for example, if when testing, it is required for the test program to issue a particular request when a certain condition occurs in a file system then using `DebugNotifySessions()`, the file system can be configured to complete a pending debug notification request whenever the condition occurs. All these methods are only available in debug builds.

