Off-Line Documentation template: clipboard support

1) Operational Goals

Provide a uniform scheme for the maintenance of the clipboard for the application, and provide an API for Document objects to get copies of the application's clip data and pass document information to the Applications clip.

2) Fundamental, "key", or cornerstone architectural requirements (POSTMORTEM)

• Do the clip conversion at the correct time, this is were the application's clip holds the same information as the desk scrap.
• provide an API to get COPIES of the DApplication::fClipData
• provide an API to put DDocument data into DApplication::fClipData

3) Model of the implementation fulfilling these key requirements (POSTMORTEM)

a) respond to OSEvent's by checking if the application is going into the foreground or background and respond by getting or giving the correct clip.
b) Provide an API to get copies of the clip in response to selecting paste from the menu.
c) Provide an API to put selected portions of documents into the application's clip in response to the user selecting copy or cut from the menu.

4) Impact/scope of the implementation on the existing body of code (POSTMORTEM)

a) DApplication::OSEvent checks if the event is a kSuspendResumeMessage or kMouseMovedMessage and checks further to see if application is going into or coming out of the background. If application is going into the background then it gives its clip to tha system (DApplication::GiveClipToSystem), otherwise it gets a copy of the systems scrap (DApplication::GetClipFromSystem).
b) DApplication::GetClipCopyFromApp returns a copy of the application's clip. The DDocument and DWindow classes may call this function in response to a paste menu selection.
c) DApplication::GiveDataToApp makes a copy of the data passed to it and places that data in the applications clip. This gets called in response to a copy or cut menu selection.

5) Coding notes (gotchas, warnings, process thoughts, items to revisited later...)

6) Testing notes(bug types, what made a bug hard to fix, what could have been done to catch it sooner....)

I was getting bit by a memory bug associated with copy and pasting. This would have been avoided if I had originally had the DDocument/DApplication clip data interface defined to make copies of the data, not references to the data. It makes copies of data whenever clip data is being passed between the DApplication and the DDocument objects. That bug was caused by improper use of the TEdit scrap.

7) Process notes (what process did you follow, could it be improved)