
Subject: Clipboard Usage

Posted by [RichardMarks](#) on Thu, 04 Jan 2024 16:40:36 GMT

[View Forum Message](#) <> [Reply to Message](#)

Greetings. I would like some assistance with getting clipboard usage sorted out.

My OS is macOS Sonoma 14.2.1 (23C71)

My Hardware is a 2020 M1 Mac mini.

I am using the U++ version downloaded from SourceForge: upp-posix-17045.tar.xz

I have a larger project in which I would like to copy information to the system clipboard for the user to paste into any other app.

I am getting the expected and undesired exclamation message box with the following minimal example.

```
#include <CtrlLib/CtrlLib.h>
```

```
GUI_APP_MAIN
```

```
{  
    if (!Upp::IsClipboardAvailableText())  
    {  
        Upp::Exclamation("Clipboard is not available unfortunately");  
    }  
}
```

I expect that the following should return true, however, that is not the case.

```
Upp::IsClipboardAvailableText()
```

Is the implementation in U++ not compatible with newer macOS?

What are my options?

Subject: Re: Clipboard Usage

Posted by [Klugier](#) on Fri, 05 Jan 2024 21:17:50 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hello RichardMarks,

I am not very familiar with Clipboard, but I can assure you that it works. I tested it by copying text

from TheIDE to Microsoft Word. I think the method you just mentioned is checking whenever top entry in clipboard is raw text or not. Please keep in mind that there are multiple functions related to clipboard (CtrlCore/CtrlCore.h file - line 1759 to 1817):

```
void ClearClipboard();
void AppendClipboard(const char *format, const byte *data, int length);
void AppendClipboard(const char *format, const String& data);
void AppendClipboard(const char *format, const Value& data, String (*render)(const Value& data));
void AppendClipboard(const char *format, const ClipData& data);
void AppendClipboard(const VectorMap<String, ClipData>& data);
String ReadClipboard(const char *format);
bool IsClipboardAvailable(const char *format);
```

```
inline void WriteClipboard(const char *format, const String& data)
{ ClearClipboard(); AppendClipboard(format, data); }
```

```
void AppendClipboardText(const String& s);
String ReadClipboardText();
void AppendClipboardUnicodeText(const WString& s);
WString ReadClipboardUnicodeText();
bool IsClipboardAvailableText();
```

```
inline void WriteClipboardText(const String& s)
{ ClearClipboard(); AppendClipboardText(s); }
inline void WriteClipboardUnicodeText(const WString& s)
{ ClearClipboard(); AppendClipboardUnicodeText(s); }
```

```
template <class T>
inline void AppendClipboardFormat(const T& object) {
    AppendClipboard(typeid(T).name(), StoreAsString(const_cast<T&>(object)));
}
```

```
template <class T>
inline void WriteClipboardFormat(const T& object) {
    ClearClipboard();
    AppendClipboardFormat(object);
}
```

```
template <class T>
inline bool ReadClipboardFormat(T& object)
{
    String s = ReadClipboard(typeid(T).name());
    return !IsNull(s) && LoadFromString(object, s);
}
```

```
template <class T>
bool IsClipboardFormatAvailable()
```

```
{  
  return IsClipboardAvailable(typeid(T).name());  
}
```

```
template <class T>  
inline T ReadClipboardFormat() {  
  T object;  
  ReadClipboardFormat(object);  
  return object;  
}
```

```
Image ReadClipboardImage();  
void AppendClipboardImage(const Image& img);
```

```
inline void WriteClipboardImage(const Image& img)  
{ ClearClipboard(); AppendClipboardImage(img); }
```

Unfortunately above code is undocumented, but you might learn a lot only from reading functions declaration. Please let me know if it helps.

Klugier
