Muse不具合一覧 - Bug #62
(V6.26)MARK文字列がテキストエリアに表示されない場合がある

2013/12/31 15:31 - Redmine Admin

ステータス:	終了	**開始日:**	2012/11/11
優先度:	通常	**期日:**	2012/11/24
担当者:		**進捗 %:**	100%
カテゴリ:		**予定工数:**	0.00時間
対象バージョン:		**作業時間の記録:**	0.00時間

説明

状況(2012.11.24)

V6.27にて対応済み。
※真因が確実に特定できたとは言えない状況ではあるが、
状況証拠に基づいた対処を行い、リリースを実施する。

(原因)
現在の所、直接の原因は不明。
間接原因としては、ビルド環境をVC6からVS2005に変更した事に因るが、
真因は別の所にあり、ビルド環境の変更が単にそれを顕在化させただけかもしれない。(2012.11.17現在)

ヒープ断片化の状態で、末尾に丁度 32 バイト分の空きがあるヒープ断片が存在していた場合、
新たなメモリ確保を行うと、VS2005 の malloc がそこを優先的に割り当ててしまう。
というのが、現時点での最有力な真因候補となっている。(2012.11.24現在)

(対処)
文字列記憶域を確保する際、デリミッタ(\0)用に１バイト追加指定していたが、これを２バイトに変更した。
プログラム構造としてはこの追加バイトは不要ではあるが、本件はカーネル障害と想定し、
NULL 終端 の次のバイトがヒープの外にならないようにするための対処として実施した。

(分析)
現象はWinXPのみに起こり、Win7では起こらない。

また、文字列が３０文字である場合に限る（30より大きくても小さくても出現しない）が、
Museはテキストエリアへの表示の見栄えを考慮し、
左マージン用に、内部で先頭に半角１文字の空白を添えている。
さらに、文字列の終端にデリミッタ(\0)の１文字が加算されるため、
実質上は、３２バイトの文字列の場合に現象が起こることになる。

・３２バイトというキリの良いサイズでのみ出現
・WinXPのみでの出現
・VS2005でのビルドのみで出現
という限られた条件下での不具合となる。

再現性は高くなく、演奏でMARK文を通過する状況を何度か繰り返すと発生する。
平均的には、３～１０回に１回程度出現するが、極めて不安定である。
また、再現するデータも比較的複雑な（あるいはリロードに時間が掛かる）演奏に限られる模様。

本現象は、MARK文だけでなく、TEXT文やSTOP文でも出現する。
また表示されていない状態で、ウィンドウの最小化と最大化を行うなど
表示のリライトを実施しても再描画されないことから
ウィンドウへの描画メッセージのタイミング問題ではない。

また、問題の文字列(MARK文の内容)を、別の手段(デバッグライトなど)で表示させた場合、
現象が起こっている際にも正しく表示されることから
NULL終端の欠落や単純なメモリ破壊などの問題ではない。

過去のバージョンをVS2005でビルドし直し、WinXPで再現テストを実施した。
V5.70のバージョン以前は、テキスト文字列の記憶域を１バイト多く取り、
それを文字列区分(TEXT/MARK/STOP)のフラグとして利用していたため、
再現データのパラメータの方を１文字減らしてみた。 これにより、同様の症状が出ることを確認した。

こうしてバージョンを遡っていくと、
結果としてソースコードが現存しているV3.1においても症状が出ることが確認できた。
更に、プログラムからあらゆるダイアログ、発音機構などを除去した状態でも再現する。
このことから、極めて基本的な部分に真因があることが予想される。

概要(2012.11.11)

●(2012/11/11) 21:33　＜himajin925さん＞

Ver6.26 にアップしました。
いろいろやってみたんですが、
（１）譜面モニタをメインウィンドと同じ幅、４／４表示にして、
（２）小節線１１０～１１３がモニタの左端に来るようにして
（３）小節線の左で右クリック
→２．戦いの…が鍵盤上部に表示されない
ようです。
画像を上げておきました。

http://tomokusaba.bne.jp/MuseWiki/index.php?plugin=ref&page=Muse%C9%D4%B6%F1%B9%E7%B0%EC%CD%F7&src=MARK1.jpg

画像を見てもらえたようですので、この現象がおきることがある、
のは分かっていただいたと思います（指揮棒カーソルは２．を過ぎているが、鍵盤の上は１．のまま）
私のところでも、同じ条件で毎回起きるわけでもないようで、？？です。
ただ、画像の場所で右クリック、２．戦いの…が表示されたら、また右クリック、
と数回やっていると高い確率で発生するようになる、らしいのですが、
初回でも起きる場合があるし、何回も起きない場合があるし、ってまったく？？です。
まったく困ってはいないのですが(^.^)

もっと前から（先頭からも？）演奏した場合も同様の場合があるようですが、
再現しない場合もあって、やはりよく分かりません。
（２）の場所が条件なのかもまったく？です（たぶん条件ではないのでしょう）

●(2012/11/11) 22:55　＜開発者＞
再現のためのレシピを忠実に何度も試したのですが、やはり私の環境では再現しませんでした。
少々厄介なことになってしまいました(苦笑)。

こういった場合考えられるのは、
(1)Museの根深く基本的なバグ
(2)メモリやリソースなどの環境の問題
のどちらかだと思いますが、
私の経験では９０％は(1)の方です。orz

ただ、再現しないことには障害追跡ができません。
本当は、こういった時こそ「純正培養データ」が欲しいのですが、

●(2012/11/12) 23:18　＜himajin925さん＞

やはり再現しませんか？
もうちょっと単純に（4/4表示で）小節線116のちょっと左で右クリックすると、
正常では、一瞬「１．ソロモンの…」が表示され、「２．戦いの…」が上書きされ、
第２曲が開始、となりますが、何回か続けてみると
（多分３回目）、「１．ソロモンの…」の表示のまま演奏が始まる、
って言う現象が、やはりおきます。

私の環境のせい？

抽出データでの再現はうまくいってません。
マクロ、第１曲の最後、第２曲～という mus ファイルを作って Museを再起動し、
演奏してみると、正常に、「２．戦いの…」が表示されます。（何度右クリックしても）
ただし、一旦元データで「１.ソロモンの…」のまま演奏される現象が発生したあと、
Museを再起動せずにこの抽出データを再生してみると……やはり「１.ソロモンの…」のまま演奏が始まったりします。

●(2012/11/12) 23:55　＜開発者＞
問題の「２．戦いの…」のMARKを、TEXTおよびSTOPに書き換えて、
それでも症状が出るかを試してもらえますか？

●(2012/11/13) 00:05　＜himajin925さん＞

TEXT → 同様です（いや、正常ケースがなくなるかな。
116の右で右クリックすると 赤字で「２．戦いの…」が確かに表示されるけど、
左で右クリックでは、いつも「１．…」のまま？）
STOP → 演奏を再開すると、「１．ソロモン…」のまま、で同様です
いずれもマクロ $1{ の中を書き換えました

●(2012/11/13) 00:34　＜開発者＞

もう一つ、実験してもらうことを思いついたので、試してもらえますか？
本来「２．戦いの…」が表示されなければならないのに「１．…」のままになっている状態で、
Museのメインウィンドウの右上にあるウィンドウの最小化ボタンを押下する。
次に、タスクバーのMuseをクリックして再びMuseウィンドウを表示させる。
これでも「１．…」のままになっているか否かを試してみてください。

●(2012/11/13) 17:39　＜そなさん＞

XP SP3なので試してみました。
再現できましたorz

●(2012/11/13) 21:05　＜開発者＞

ご報告、ありがとうございます。
Museのバグである公算が高まってきたのは残念ですが、
再現する試験者が増えたのは心強い限りです。
お手数ですが、お手すきの時に以下の実験をして結果を教えてもらえますか？
＝＝＝＝＝
本来「２．戦いの…」が表示されなければならないのに「１．…」のままになっている状態で、
Museのメインウィンドウの右上にあるウィンドウの最小化ボタンを押下する。
次に、タスクバーのMuseをクリックして再びMuseウィンドウを表示させる。
これでも「１．…」のままになっているか否かを試してみてください。

●(2012/11/13) 21:18　＜himajin925さん＞

（１）「最小化→元に戻す」または「他のウィンドで隠す」いずれも、
　　「演奏中」「演奏停止」どっちでも、「１．ソロモンの…」のままです。
（２）次いで、文字列そのものを疑っていろいろやってみました。
文字列の中身をまず疑ったのですが（全角スペース＋D など）どうも違うようで（４．饗宴の踊り…では発生しない）
そのスペースを半角に置き換えたりしてみているうちに、どうも「３０バイトの文字列が指定された場合に発生する
のではないか」という推測に達しました。

*MARK"１２３４５６７８９10百千万億兆"
*MARK"0123456789012345678901234567 89"
*MARK" 2.戦いの踊り　Danza guerresca"

↑全角・半角の違いが分かりにくいですが…いずれも３０バイトのはずです
この３ついずれでも発生します。また「４．饗宴の踊り…」でも半角文字を１つ削除すると、
やはり発生します（３０バイトになります）。
３０バイト以下と超える場合で処理が分かれる、など何かありますか？

●(2012/11/14) 09:08　＜そなさん＞

> 本来「２．戦いの…」が表示されなければならないのに「１．…」のままになっている状態で、
> Museのメインウィンドウの右上にあるウィンドウの最小化ボタンを押下する。
> 次に、タスクバーのMuseをクリックして再びMuseウィンドウを表示させる。
> これでも「１．…」のままになっているか否かを試してみてください。

やってみました。

再生中 停止中 ともに「１．…」のままでした。
起動直後の演奏では正常に切り替わることが多い気がします。

ついでに 30バイト説も実験
$2{*MARK" 2.戦いの踊り　Danza guerresca"}→元のまま→発生
$2{*MARK" 2.戦いの踊りDanza guerresca"}→スペースを削除→正常に表示
$2{*MARK" 2.戦いの踊り　Danza guerresca+"}→最後に+を追加→正常に表示
$2{*MARK"0123456789012345678901234567 89"}→発生
$2{*MARK"0123456789012345678901234567 89+"}→正常
$2{*MARK"0123456789012345678901234567 8"}→正常

書く位置を変えてみる
${2}を 1小節前に書く→発生
${2}を8分音符分後ろに書く→正常

こんな感じでしたが お役に立つでしょうか〜？

追記

ver
5.82 発生せず
5.92 ↓
6.01 ↓
6.03 発生せず
6.21 発生

109小節目の頭から再生すると発生率高め？
クラのロングトーンがあるから？

MARKをSTOPに書き換えて
> 本来「２．戦いの…」が表示されなければならないのに「１．…」のままになっている状態で、
> Museのメインウィンドウの右上にあるウィンドウの最小化ボタンを押下する。
> 次に、タスクバーのMuseをクリックして再びMuseウィンドウを表示させる。

すると 色のみ赤に変化

●(2012/11/14) 10:56　＜H.N.WPKIDSさん＞

Windows 7 上で動くWindows XP Mode で症状を確認しましたのでご連絡します。
当該症状のにつきまして、同様の結果が出ました。
こちらは再現率が高い *MARK を *STOP に変えた時のスクリーンショットとシステムのプロパティ画面です。
なお、互換モードでXP SP3を動かした時では発生しませんでした。

http://tomokusaba.bne.jp/MuseWiki/index.php?plugin=ref&page=Muse%C9%D4%B6%F1%B9%E7%B0%EC%CD%F7&src=MARK2.png

●(2012/11/14) 12:54　＜木下さん＞

再現性は１００％ではないですが，症状が出ていますので報告します。

ＯＳ：Windows2000 SP4

マークの文字列が更新されない条件
・「シバの女王ベルキス」MARK"2. "とMARK"4. "
・譜面モニタでマークの前の位置でリロードする
・マークの文字列は丁度３０文字（全角文字は２と数える）

状況
・１度更新されないと，その後の更新されない確立がかなり上昇する
・シーク開始位置で確立が変化する
・譜面モニタ１１４の２拍目の確立が高い
・確立が１００％や０％の場所は無い
・↑でのリロード＆演奏でも再現

関係が不明
・文法エラー表示

自分のデータで上記の条件で再現しないか調査中ですが未だに再現していません。
いま少し調べてみます。

> 本来「２．戦いの…」が表示されなければならないのに「１．…」のままになっている状態で、
> Museのメインウィンドウの右上にあるウィンドウの最小化ボタンを押下する。
> 次に、タスクバーのMuseをクリックして再びMuseウィンドウを表示させる。

表示は更新されないままでした。

●(2012/11/14) 15:46　＜Elekenさん＞

少し気になったので、私の環境 (Windows XP SP3) でも調べてみました。
気付いた点は以下の３点です。

(1) 発生条件について
(a) データのロード(おそらく構文解析かテンポ処理)に時間がかかる場合
(b) *MARK や *TEXT 構文などで、テキスト内容が丁度 30 バイトの場合
(a) かつ (b) の条件のとき、構文の直前からリロード再生を行うと発生するようです。
たとえば、以下のデータで、譜面モニタ上の最初の小節で何度か右クリックすると再現します。

*HEAD"This is a dummy." _1
*MARK"1234567890123456789012345678 90" _4
{%{#A0 v10%80:16@V-20m4%90V+20r}{#B0 v10@V-20m4V+20r}{#D0 v10@V-20m4V+20r}} (←この行を 2000 回繰り返す)

また、単にファイルサイズが大きいだけではだめで、構文がある程度複雑でないと再現しないようです。

(2) 発生条件を一度満たした後の Muse の挙動について
一度この現象が発生した Muse で、現象が再現しないはずの他のデータを読み込んだ場合でも、
(b) の条件を満たしただけで同じ現象が発生します。
このことから、この現象は Muse の内部状態を変化させていることが示唆されます。

(3) 現象が発生する Muse のバージョンについて
最近の古いバージョンは持っていないのですが、Ver. 2010 では発生せず、
Ver. 2012 では既に発生することを確かめました。
よって、この間の更新が現象の一因になっていることが示唆されます。

●(2012/11/17) 16:30　＜himajin925さん＞

SetWindowText なら、戻り値はどうでしょう？（ GetLastError は有用な情報は返してくれない？でしょうが）

●(2012/11/17) 17:55　＜開発者＞
早速試してみたのですが、結果は TRUE でした。orz
再現性は確保できているので、科学的にアプローチできます。
今は、履歴データによって現象が出たり出なかったりする件を追っていました。
これもまた微妙な按配です。
ほんの少し履歴データを変更すると、途端に現象が出なくなります。
Museデータを、ちょっち変更すると出なくなるのと同じ印象です。
まるで、ガラス細工のようです(苦笑)。

で、再現する状態でプログラムを少しずつ削り取っていって、
症状が出なくなる瞬間を把握しようと思って作業していたのですが、
結局、MuseWikiの（対処）の項に記載したのと同じ結果に行き着きました。

テキストエリアに表示する文字列のメモリ確保を１バイト増やすか、
履歴データのパス文字列のメモリ確保を１バイト増やすか、
どちらか一方でも行うと症状は出なくなります。

アライメントの関係かとも思い、
コンパイルの最適化オプションを外すなどもしてみたのですが、

結果は同じでした。

＞ＯＳ依存の現象のようで、大変そうです(-;)が、お待ちしてます
現時点の対処方法(１バイト余分にメモリ確保)は、今一つ釈然としないのですが、
一応今まで追跡した内容からして、
限りになく患部に近い所に薬を塗れているのではないか、
と思い始めてきました。

●(2012/11/18) 14:59　＜Elekenさん＞

個人的にはとても原因が気になります。
以下は Ver. 6.26 で試した結果です。

まず、デバッガ(Spy++)でウィンドウメッセージをトレースしたところ、問題が生じるタイミングでは
そもそも WM_SETTEXT メッセージが発行されていないようです。
次に、同一タイミングでたくさんの 30 バイトの MARK コマンドを発行するようなデータ
（ *MARK"MARK 01_____" の数字部分を 01 ～ 32 まで変える）
で試したところ、

・いくつかの MARK コマンドによる WM_SETTEXT が発行されない
・どの MARK コマンドが表示されないかは、リロードされるたび変わる
・逆に、リロードしない限りは、先頭から再生を繰り返しても、表示されない MARK コマンドの種類は変わらない

という結果になりました。
問題はリロード時に生じているのではないかというのが、私の推測です。

●(2012/11/18) 16:41　＜開発者＞
＞ 問題が生じるタイミングではそもそも WM_SETTEXT メッセージが発行されていないようです。
＞ 問題はリロード時に生じているのではないかというのが、私の推測です。
これは、かなり核心を突いているかもしれません。
この仮説に基づいて、再度追跡してみます。

ちなみに、
＞ 以下は Ver. 6.26 で試した結果です。
ということは、V6.26aでは、やはり症状は抑え込まれているということでしょうか？
＞ ・逆に、リロードしない限りは、先頭から再生を繰り返しても、表示されない MARK コマンドの種類は変わらない
もしかしたら、この状態でMIDIファイルのエクスポートをしたら、
そのMIDIファイルも、当該のMARK文が欠落した状態かもしれませんね。
Museは演奏データとMIDIエクスポートデータを一元化しているので、
MIDIエクスポートデータは、そのまま演奏データのダンプデータとして活用できます。

●(2012/11/18) 21:56　＜Elekenさん＞

> ということは、V6.26aでは、やはり症状は抑え込まれているということでしょうか？
そうです。今のところ、V6.26a では症状は生じていません。
ちなみに、V6.26 を互換モードで起動した場合では、Windows 2000 以外のモードでは症状は生じません。

> MIDIエクスポートデータは、そのまま演奏データのダンプデータとして活用できます。
なるほど！早速試してみました。
症状が出た状態で MIDI エクスポートをしても、正常なデータが出力されるのみです。
ということは、メモリ上のデータは全て正常？？
OS 依存ということも考えると、私の考えていたより問題の根は深いのでしょうか・・・

●(2012/11/18) 16:41　＜開発者＞
＞ OS 依存ということも考えると、私の考えていたより問題の根は深いのでしょうか・・・
多分、メモリ上のデータは全て正常だと思います。
しかし、どこかでMuseがデータ以外のメモリを壊していることが考えられます。
症状はMARK表示の演奏時に出ますが、Elekenさんが推察されているとおり、
リロード（あるいはロード後の後処理）で既に破壊工作は完了しているのではないかと(苦笑)。

ちなみに、
＞ ではそもそも WM_SETTEXT メッセージが発行されていないようです
の件を追試してみましたので、ご報告します。
SetWindowText()をコールしている箇所で、デバッグライトにて
・SetWindowText()の戻り値

・SetWindowText()で送信している文字列内容
を確認してみました。

なんと面妖なことに、Spy++でメッセージが送られていない状況でも、
・戻り値は、TRUE
・文字列は正常な３０文字
が確認されました。
つまりデータは壊れていない、ということを裏付けていると思います。

Elekenさんのご指摘の通り、演奏時点での症状から追跡するのを止めて、
データロード時点の不具合を追跡してみようと思います。
(2012/11/14) 15:46 でElekenさんが作ってくれた「純正培養データ」の純度を更に高めることができました。

```
*HEAD"This is a dummy." _1
*MARK"1234567890123456789012345678 90" _4
%
{{#A0 m4r}{#B0 m4r}{#D0 m4r}}
{{#A0 m4r}{#B0 m4r}{#D0 m4r}}
{{#A0 m4r}{#B0 m4r}{#D0 m4r}}
・
・
・
(↓この行を 2000 回繰り返す)
```

です。
結局、属性関係は無くても症状が出ますので、譜面モニタの属性表示はシロだったみたいです。
また、マクロの繰返し数で2000回指定をすると症状が出ません。
どうも、無名マクロの処理にバグが混入している気がしてきました。
今度こそ、犯人の潜むアジトを特定できるといいのですが・・・。

●(2012/11/18) 16:41　＜開発者＞
経過報告です。
＞ (2012/11/14) 15:46 でElekenさんが作ってくれた「純正培養データ」の純度を更に高めることができました。

```
*HEAD"This is a dummy." _1
*MARK"1234567890123456789012345678 90" _4
%
#A0
{ d }
{ d }
{ d }
・
・
・
(↑この行が多い程、再現確率が高まる。10,000行あれば楽勝で出現)
```

だいぶ敵陣に近づいてきたような気がします。
ただ、無名マクロの処理はとっても複雑なので、気が重いです。(苦笑)

●(2012/11/23) 03:26　＜Elekenさん＞

失礼ながら実行時の Muse(ver.6.26) をメモリダンプして解析したところ、次のことが分かりました。
＊メモリ上に確保された MARK 文のテキストの最初の文字 (半角スペース) から見て、オフセット -3 バイト目のメモリが、
・正常に表示されるテキスト -> 0x01
・表示されないテキスト -> 0x11
になっている
このアドレスのメモリが何を表すのか分かりませんし、他の環境でどうなるかもわかりませんが、この共通点は少し怪しそうです。

●(2012/11/24) 00:16　＜himajin925さん＞

SendMessage （SETTEXT) で送るメッセージが HEAP の末尾に来た場合、メッセージが飛んでいないようです。
なぜそうなるのでしょう？

Win95、98、Me ではおきず、XP 2000 でおきるというと、SendMessage の変換　（UNICODE 関連）のあたりの
仕様の違いに関連ありそうとは思いましたが、これはという情報はまだ見つかってません。

症状が一定しないのは、きっと単なるリロードでは HeapCreate しなおしていない？のではないか、と思いました。
前のコンパイルの HeapAlloc したメモリが残っている、などないでしょうか？

32バイトちょうどだとなぜ末尾に来るのか？
（もしかすると、このメッセージの分を HeapAlloc する際、拡張が起きているのかな？）
なぜ末尾ではいけないのか？←これが問題ですね。まだまだ分かりません。

ただ、どうも「アプリのバグ」という感じではない、と思います

多分 私の環境でも、release ビルドだったら、HeapAlloc したときの保護バイトが付かないだろうから（？）、
Heap の最後に確保されるように工夫して、SendMessage してみようと思います。

●(2012/11/24) 02:20　＜Elekenさん＞

> SendMessage （SETTEXT) で送るメッセージが HEAP の末尾に来た場合、メッセージが飛んでいないようです。
なるほど！
謎(の一部)が解けました。
つまり、昨日の投稿でのオフセット -3 バイト位置は、HeapAlloc でできたメモリ管理用構造体で、
0x01 は空きあり、0x11 はヒープ末尾を表すフラグだったわけです。

この NULL 終端がヒープ末尾に来る文字列の問題は、以下の簡単なプログラムで確認できます。

```
HANDLE hHeap = HeapCreate(0, 16384, 16384);
int i, ret, cnt = 0, alloc_bytes = 32;
char* p_last, * p_first, *p;

p = p_first = (char*)HeapAlloc(hHeap, 0, alloc_bytes);

while(1){
    p_last = p;
    sprintf(p, "Allocation #%04d._____________", cnt++); // 終端込みで 32 バイト
    if((p = (char*)HeapAlloc(hHeap, 0, alloc_bytes)) == NULL) break;
}

HWND hWnd = (HWND)0x001F0A78; // 適当なウィンドウ

for(i=0; i<10; i++){
    _sleep(1000);
    ret = SendMessage(hWnd, WM_SETTEXT, 0x00, (LPARAM)p_first); // 表示される
    printf("text= %s, ret = %d\n", p_first, ret);
    _sleep(1000);
    ret = SendMessage(hWnd, WM_SETTEXT, 0x00, (LPARAM)p_last); // 表示されない
    printf("text= %s, ret = %d\n", p_last, ret);
}
HeapDestroy(hHeap);
```

必ずしも 32 バイトの HeapAlloc が問題なわけではなく、HeapAlloc を呼び出した時点の残り割り当て可能領域に依存します
。
例えば上記プログラムなら、こちらの環境 (WinXP, コンパイラ VC++6.0) では、少なくとも
(ヒープ領域サイズ, 割り当てサイズ) = (16384, 32), (16384, 24), (8192, 24), (8192, 16) のときに発生します。

スレッドをまたいで WM_SETTEXT を送信するので、カーネルがテキストをコピーする必要があり、
そのときにカーネルのバグで、NULL 終端を越えて走査しようとしてしまう・・・というストーリーでしょうか？

●(2012/11/24) 11:09　＜開発者＞
＞ HeapAlloc を呼び出した時点の残り割り当て可能領域に依存します。
思い当たる節があります。
純正培養Museを作る過程で、本来必要なFree()部分をコメントアウトしたら、症状が出なくなりました。
また、機能を削り取ることで不要となった構造体のメンバをコメントアウトしたら、症状が出なくなったこともありました。

●(2012/11/24) 17:36　＜Elekenさん＞

あまり自信はないですが、
複雑なデータをロードした後の状態が、ヒープの断片化を起こしていて、
リロードした後、テキスト構文のメモリ確保の段階で
末尾に丁度 32 バイト分の空きがあるヒープ断片が存在して、
VS2005 の malloc がそこを優先的に割り当ててしまうということなのでしょうか。

もしカーネルのバグだとすると、それ以外のライブラリは全て仕様内の動作をしているので、解決策は
・SendMessage で送信する文字列の NULL 終端 の次のバイトがヒープの外にならないようにする
という一点で、そのためには、
(1) 文字列領域は 1 バイト多く確保する
(2) 別に用意したヒープ末尾でないバッファに文字列を一旦コピーして、それを SendMessage する
(3) ヒープ断片化が起こらないように、ヒープは自分で管理する。リロードの度に HeapDestroy -> HeapCreate

…のどれかをとればいいように思います。

●(2012/11/24) 20:01　＜開発者＞
解決策のご提示、ありがとうございます！
＞ (1) 文字列領域は 1 バイト多く確保する
これは、まさにV6.26aの対応そのものですね。
＞ (2) 別に用意したヒープ末尾でないバッファに文字列を一旦コピーして、それを SendMessage する
今回の格闘の最中、実はこれを実施した記憶があります。
SendMessage()する前に、別途確保した文字列変数に問題の文字列内容をコピーし、
それをSendMessage()に送ってみました。すると、症状は出ませんでした。
ただ、演奏時点の処理負荷が増えるので、対応策としては却下していました。
場当たり的に試行した私の実験と、論理的に導いたElekenさんの対応策の結果が見事に一致していますので、
Elekenさんの今回の真因推理はかなりの確度で正しいと思えます。
＞ (3) ヒープ断片化が起こらないように、ヒープは自分で管理する。リロードの度に HeapDestroy -> HeapCreate
私の技術力からして、この対応は継続的にメンテナンスしていく地震がありません。
じゃなかった、自信がありせん。そういえば先程、横浜で比較的大きな自身がありました。
じゃなかった、地震がありました。
＞ …のどれかをとればいいように思います。
という訳で、「(1) 文字列領域は 1 バイト多く確保する」を正式な対応とすることが、
現時点では濃厚です。

●(2012/11/24) 21:53　＜himajin925さん＞

32Byte で発生するなら、なぜ他の 8 の倍数バイトで発生しないのかも謎です。
それとも、mus ファイルの内容によっては、（見つけていないだけで）発生する場合があるのかな？

●(2012/11/25) 02:23　＜Elekenさん＞

＞ 32Byte で発生するなら、なぜ他の 8 の倍数バイトで発生しないのかも謎です。
＞ それとも、mus ファイルの内容によっては、（見つけていないだけで）発生する場合があるのかな？

それが一番の謎です。

ヒープ領域固定の場合、32 byte の malloc がヒープ終端に来るケースは、
malloc(HeapAlloc) のオーバーヘッドが 8 byte なので、ヒープ残量 40 byte のときです。
この状態で、30 byte *TEXT の代わりになるような、*TEXT 文の組み合わせも存在しそうに思います。

このときに連続して malloc すると問題になるパターンは、
(32 byte), (1-8 byte -> 16byte), (13-16byte -> 8 byte) の 3 パターンです。
しかしながら、30byte *TEXT の代わりに (6byte *TEXT -> 14byte *TEXT) などとしても、問題は起こりません。

Muse でのケースは、ヒープ断片化を起こしていると考えられるので、より複雑です。
つまり、malloc 関数には「末尾 40 byte 領域」をアロケートする他に、別の領域を使う選択肢があるからです。
（他に、ヒープを広げるという選択肢も！）
malloc 関数は 前述の状態において、24 byte よりも小さい領域が指定されたとき後者を選択し、
25-32 byte が指定されたときのみ、前者を選択するのではないでしょうか。（メモリ効率上、賢い戦略だと思います）

このあたりは、malloc のソースコードを見ないとわからないので、全く確証が持てません。

なぜ「32」byte かは、もし仮にマクロ展開時に作る構造体が 25-32 byte なら

説明がつきそうですが・・・。

●(2012/11/25) 09:37　＜himajin925さん＞

しつこいなぁと思われたでしょうが(^_^;)
これで、「Muse にバグはなかった！」で良いでしょうね。よかったよかった

（１）純粋培養 Muse の様子を見ていて、症状の起きない 95-Me の互換モードでは、XP でのメモリ管理と違う
（ヘッダを見ると、95-Me はバイト単位での割り付けかと想像される→別の API で割り付けてる？）ので問題がおきない？
（２）XP 以降では 8バイト単位の割り付けになるが、開放されたブロックの再利用の仕方
（同じ大きさの開放領域があればそれを使うが、その探し方）が　Vista 以降で変更になったそうで、
そのせいで XP でだけ発生するのでしょうか。ただ、そうだとすると同様の条件になるケースはありそうだと感じます。
XP で顕在化しただけで、Win7 でも…いや何らかの修正がされているのかな？
（３）ちょうど 40 バイト空いていた場合だけではなくて、
ちょうど 16バイト空きで 8バイトの文字列…
ちょうど 24バイト空きで 16バイトの文字列…
…
なんていうことが、データによってはありそうですが…そのうち見つかる可能性もあるのかな

で、やはり +1 とするのですか？

●(2012/11/25) 11:09　＜開発者＞
＊Elekenさん
＞ なぜ「32」byte かは、もし仮にマクロ展開時に作る構造体が 25-32 byte なら
＞ 説明がつきそうですが・・・。
Elekenさんって、まるでシャーロックホームズみたいですね！
私が掲示板(2012/11/24) 11:09 で書き込んだ、
＞ また、機能を削り取ることで不要となった構造体のメンバをコメントアウトしたら、症状が出なくなったこともありました。
を具体的にお示しします。何か考察のヒントになるかもしれません。
既に純正培養Museでは３つの構造体しか使用しておらず、
利用しないメンバーもどんどん剥ぎ取っていますので、
以下の様な簡素な内容になっています。

```
// ノート構造体 -------------------------------
typedef struct t_note {
    struct t_note* mpz_next;    /* 次ポインタ */
    char* mpc_txt;            /* コマンド文字列 */
    DWORD mh_tim;             /* イベント時刻(ミリ秒) */
    char mc_mid;            /* MIDIデータ */
} T_note;

// データ展開ワーク ----------------------------
typedef struct t_work {
    struct t_work* mpz_next;    /* 次ポインタ */
    char* mpc_data;            /* データ文字列 */
    int mi_num;            /* ★行番号 */
} T_work;

// 定義マクロリスト ---------------------------
typedef struct t_macr {
    struct t_macr* mpz_next;    /* 次ポインタ */
    char* mpc_name;            /* マクロ名(無名マクロの場合はNULL) */
    T_work* mpz_p0;            /* ★開始アドレス( { の次) */
    T_work* mpz_p1;            /* ★終了アドレス( } の前) */
    char mc_ext;            /* ★展開中フラグ */
} T_macr;
```

この中の★印の付いたメンバーは、現時点の純正培養Museでは使用していません。
そこでこれらの★付きメンバーも剥ぎ取ろうとしたのですが、１つでも取り払うと症状が出なくなってしまいました。
＊himajin925さん

＞ しつこいなぁと思われたでしょうが(-;)
いえいえ、むしろ“ 真実を追究する ” 真摯な姿に感動さえしています。
少々大袈裟かもしれませんが、
この姿勢は、科学者や検事や医師などの職業を問わず、
人のあらゆる活動で尊重されるべきものだと思います。
＞ で、やはり +1 とするのですか？
はい。可変長文字列のmalloc()だけ１バイト余分に確保するようにしようと思います。
ほとんどは、コンパイル時の一時的な確保ですから、永続的にメモリ効率を落とす訳ではないし、
演奏時にも確保し続けるのは、テキストの表示文字列と譜面モニタの属性表示文字列、
履歴データのパス、楽譜出力用のタイトル群とスルーコマンド位です。
そもそもたった１バイトですので、大勢に影響は無いでしょう。

※その対応箇所のコメント文には、今回の趣旨が分かるように書いておこうと思います。
さもないと１０年後の自分が見直した時、無駄なメモリを確保している！
と、戻してしまいそうです(笑)。
＞ これで、「Muse にバグはなかった！」で良いでしょうね。よかったよかった
とても安堵しています。＼(º)／
でもこの“ 戦いの踊り ” が終わり、お二人とエキサイティングなやりとりもできなくなると思うと、
ちょっち淋しい感じもしたりして(苦笑)。
記念といってはなんですが、現時点の純正培養Museをアップデートしておきました。

●(2012/11/25) 02:23　＜Elekenさん＞

> 機能を削り取ることで不要となった構造体のメンバをコメントアウトしたら、症状が出なくなったこともありました。

メモリをみたところ、「末尾 40 byte 領域」を持つヒープ断片は
<24 バイト, <16 バイト, <8 バイトのデータが規則正しく埋まっているようです。
ご提示頂いた構造体と照らし合わせると、前２者は t_macr, t_work であるようです。
（ malloc ではサイズが 8 バイト単位に切り上げられます）
このへんの絶妙なバランスが、「末尾 40 byte 領域」を導いていたと推測しています。

しかし、すみません、折角ご提示頂いたのですが、これ以上の探求は
ライブラリの malloc, free の解析を伴うので難しいです。
これでゲームクリアとさせていただきます（笑）。

himajin925 さんの仰るような未確認バグデータに対応する意味も含めて、
+1 バイト余計に確保することはスマートな解決策だと思います。
malloc １回で 8〜15 バイトの無駄ですから、1 バイトのオーバヘッドは誤差のようなものだと思います。

> 記念といってはなんですが、現時点の純正培養Museをアップデートしておきました。
このソフトウェアが "muse.exe" を名乗っているのを見ると、Muse のアイデンティティってなんだろうと暫し考えてしまいます。
テキスト領域と鍵盤ひとつあれば Muse なのかな？

ともあれ、Muse は思っていた以上によく考えられて設計されているのだと、
ただただ感服するばかりです。
今回は難しいパズルを見つけた気がして、少し書き込みすぎてしまいました。
一番楽しんでいたのは私だったかも知れません。

関連するチケット:

関連している Release # 160: Muse V6.26	**終了**	**2012/10/27**

履歴

#1 - 2013/12/31 15:31 - Redmine Admin

- 説明 を更新