Subject: SetLineKeyValue - in memory
Posted by luoganda on Fri, 21 Apr 2017 15:55:06 GMT
View Forum Message <> Reply to Message

Empty lines and (// or ;) at start of line are ignored
chkForEqualSign:
  when true - there must be '=' after key(even if it's empty)
  when false - line can contain only a key(withNo=), but it's added before fn returns

NewLine: \n, \r and \r\n are accepted

Update:
  -func now accepts key that has some kind of header,eg SetLineKeyValue(lines,"key","value");
while
   lines has somewhere 'keyAnyHdr'(eg keyLine1\nLine2=value, or key:flg:20,flg2:coffee=value)
  -added 2helperfuncs for pack 2vals in value - chk down code - can be used eg with
WithDropChoice<EditString>

```
bool SetLineKeyValue(String& lines,const String& key,const String& val,dword startAt=0,bool
chkForEqualSign=true,dword *chkForExistanceOnly=0,String *prevBack=0){
  if(!lines.GetCount())return false;
  ASSERT(key.GetCount());
  ASSERT(startAt<lines.GetCount());
  register char *p=(char*)~lines+startAt;
  int64 fpos; char *pst;
  int pos,off,linenum;
  if(prevBack)*prevBack="";
  while(*p){
    fpos=p-~lines; linenum=0;pos=0;pst=p;

while(*p){if(*p=='\r'&&*p=='\n'){linenum=p-pst;break;}if(*p=='\r'||*p=='\n'){linenum=p-pst;break;}if(*p
=='=')pos=p-pst; ++p;}
    if(*p)++p;
    if(linenum==0 && 0==(linenum=p-(*p?1:0)-pst))continue;
    if(linenum&&*pst==';'||linenum>=2&&pst[0]=='/'&&pst[1]=='/')continue;
    if(chkForEqualSign&&!pos)continue;

if(linenum<key.GetCount()+(chkForEqualSign?1:0)||0!=memcmp(pst,~key,key.GetCount()))contin
ue;

if(prevBack&&linenum>pos/*key.GetCount()*/+1)*prevBack=lines.Mid(fpos+pos/*key.GetCount()*/
+1,linenum-(/*key.GetCount()*/pos+1));
    if(chkForExistanceOnly){*chkForExistanceOnly=(dword)(int32)fpos;return true;}
    fpos+=/*key.GetCount()*/pos+1;
    off=(/*key.GetCount()*/pos+1+val.GetCount())-linenum; //speedup a litle - don't insert/remove
```

```
twice
    if(off<0)lines.Remove(fpos,abs(off)); else if(off)lines.Insert(fpos,"R",off);
    *(char*)lines.GetIter(fpos-1)='='; memcpy((void*)lines.GetIter(fpos),~val,val.GetCount());
    return true;
    }
  /*
  StringStream ss(lines);
  while(!ss.IsEof()){
    fpos=ss.GetPos();
    line=ss.GetLine();if(chkForEqualSign&&-1==(pos=line.Find('')))continue;
    if(!line.StartsWith(key+(chkForEqualSign?"=":"")))continue;
    if(pos==-1){insEq;pos=key.GetCount();}
    off=(key.GetCount()+1+val.GetCount())-line.GetCount(); //speedup a litle - don't insert/remove
twice
    fpos+=key.GetCount();
    if(off<0)lines.Remove(fpos,abs(off)); else if(off)lines.Insert(fpos," ",off);
    *(char*)lines.GetIter(fpos)='='; memcpy((void*)lines.GetIter(fpos+1),~val,val.GetCount());
    //lines
    break;
    }*/
  return false;
}


//when editor that edit this can't handle fe \1 chars, use something else
//errline is just an errorlike func PromptOK can be used instead of it

inline String SLKVValue2Pack(const String& val1,const String& val2,String withCh="\1"){return
val1+withCh+val2;}
static String SLKVValue2Get(const String& val,String *val1=0,const String& sepCh="\1",bool
mustBeSepCh=false){
  /*if(val1)*val1="";ifval1==in*/int i=val.ReverseFind(sepCh);
  if(i==-1){if(mustBeSepCh){errline(el,"could not find SLKVValue2Get
sepChar");if(val1)*val1="";return String::GetVoid();}if(val1)*val1=val; return "";}
  String ret=val.Mid(i+sepCh.GetCount());
if(val1)*val1=val.Mid(0,i&&sepCh.GetCount()==1&&val[i]=='\n'&&val[i-1]=='\r'?i-1:i); return ret;
}
```

Subject: Re: SetLineKeyValue - in memory
Posted by luoganda on Mon, 24 Apr 2017 08:42:24 GMT
View Forum Message <> Reply to Message

These can be used when info is known to be contignous,
errline is just an errorlike func PromptOK can be used instead of it
All are readOnly funcs

```
inline bool SLKVSkipEmpty(StringStream &ss){
  bool is=0;

while(!ss.IsEof()&&(ss.Peek()=='\n'||ss.Peek()=='\r'||ss.Peek()==';'||*(word*)ss.PeekPtr(2)==*(word
*)"//")){
    is=1;ss.GetLine();
  }
  return is;
}

inline String SLKVKey(StringStream &ss,bool withEqSign=true){
  String ln=ss.GetLine();
  int i=ln.Find('=');if(i==-1){if(withEqSign){errline(el,"char '=' not found near %s",~ln);return
String::GetVoid();}}else ln=ln.Mid(0,i);
  return ln;
}

inline String SLKVValue(StringStream &ss){
  String ln=ss.GetLine();
  int i=ln.Find('=');if(i==-1){errline(el,"char '=' not found near %s",~ln);return String::GetVoid();}
  return ln.Mid(i+1);
}

inline bool SLKVKeyValue(StringStream &ss,String& key,String& val,bool withEqSign=true){
  key=val="";String ln=ss.GetLine();
  int i=ln.Find('=');if(i==-1){if(withEqSign){errline(el,"char '=' not found near %s",~ln);return
false;}key=ln;return true;}
  key=ln.Mid(0,i); val=ln.Mid(i+1);
  return true;
}
```

Subject: Re: SetLineKeyValue - in memory: smallBugFix
Posted by luoganda on Tue, 25 Apr 2017 19:43:31 GMT
View Forum Message <> Reply to Message

Modified upper SLKVKey(from ln.Mid(i) to ln.Mid(0,i))
now returns right result...

Subject: Re: SetLineKeyValue - in memory: bugFix
Posted by luoganda on Tue, 25 Apr 2017 23:21:46 GMT
View Forum Message <> Reply to Message

SetLineKeyValue,

now returns right result...

Subject: Re: SetLineKeyValue - in memory: update
Posted by luoganda on Sun, 30 Apr 2017 11:25:00 GMT
View Forum Message <> Reply to Message

update -> check original message

Subject: Re: SetLineKeyValue - in memory:update2
Posted by luoganda on Sun, 30 Apr 2017 12:33:53 GMT
View Forum Message <> Reply to Message

Using SetLineKeyValue with StringStream:
-StringStream: a few funcs should be added to use this,chk down

If you read this, there was also original func update - chk original message

Usage:
-to speedup and data(lines) is/are known to be contignous,then;
  SetLineKeyValue(ss,"a key","a value");
  ss.GetLine(); //advance pos to next line,so next is found immediatelly
-when keys(data) is not known to be contignous(butAreSearchable),then;
  SetLineKeyValue(ss,"a key","a value"); //just this

```
bool SetLineKeyValue(StringStream& ss,const String& key,const String& val,bool
chkForEqualSign,dword *chkForExistanceOnly,String *prevBack){
  if(ss.IsEof())return false;
  register char *p=(char*)ss.PeekPtr();
  int64 fpos=ss.GetPos();ss.Seek(0);
  char *pst,*porg=p,*ssSt=(char*)ss.PeekPtr();
  int pos,off,linenum;
  ss.Seek(fpos);
  if(prevBack)*prevBack="";
  while(*p){
    fpos=ss.GetPos(); linenum=0;pos=0;pst=p;

while(*p){if(*p=='\r'&&*p=='\n'){linenum=p-pst/*p-~lines-fpos*/;break;}if(*p=='\r'||*p=='\n'){linenum=p
-pst/*p-~lines-fpos*/;break;}if(*p=='=')pos=p-pst/*p-~lines-fpos*/; ++p;}
    if(*p!=0)++p;
    if(linenum==0 && 0==(linenum=p-(*p?1:0)-pst/*~lines-fpos*/))continue;
    if(linenum&&*pst==';'||linenum>=2&&pst[0]=='/'&&pst[1]=='/')continue;
    if(chkForEqualSign&&!pos)continue;
```

```
if(linenum<key.GetCount()+(chkForEqualSign?1:0)||0!=memcmp(pst,~key,key.GetCount()))contin
ue;if(chkForEqualSign&&pst[key.GetCount()]!='=')continue;

if(prevBack&&linenum>fpos+/*key.GetCount()*/pos+1)*prevBack=String(pst+/*key.GetCount()*/po
s+1/*lines.Mid(fpos+key.GetCount()+1*/,linenum-(/*key.GetCount()*/pos+1));
    if(chkForExistanceOnly){*chkForExistanceOnly=(dword)(int32)fpos;return true;}
    fpos+=/*key.GetCount()*/pos+1;
    off=(/*key.GetCount()*/pos+1+val.GetCount())-linenum; //speedup a litle - don't insert/remove
twice
    if(off<0)ss.Remove(abs(off),fpos); else if(off)ss.Insert('R',off,fpos);
    *(char*)ss.GetIter(fpos-1)='='; memcpy((void*)ss.GetIter(fpos),~val,val.GetCount());
    //if(off<0)slns. lines.Remove(fpos,abs(off)); else if(off)lines.Insert(fpos," ",off);
    //*(char*)lines.GetIter(fpos-1)='='; memcpy((void*)lines.GetIter(fpos),~val,val.GetCount());
    return true;
    }
  return false;
}
```

StringStream added code for this - but should be revisited;
StringStream has Put methods but no Remove(Pos isTakenInConsideration), here they are - but are not yet 100% tested
optimally it should be added between SetSize and Open

```
 virtual  void  Insert(int ch,int num=1,int at=-1);
 virtual  void  Insert(const String& str,int at=-1);
 virtual  void  Remove(int num=1,int at=-1);
 virtual  const byte* GetIter(int at){ASSERT(at<GetSize());return buffer+at;}

//cpp files
void StringStream::Insert(int ch,int num,int at){
 Insert(String(ch,num),at);
}
void StringStream::Insert(const String& str,int at){
 if(at==-1)at=GetPos();
 ASSERT(at+str.GetCount()<=GetSize());
 SetReadMode(); //justToKnowWhichToUse...
 int64 npos=at<GetPos()?GetPos()+GetPos()-at:GetPos();
 data.Insert(at,str);
 buffer = (byte *) ~data;  //<=this should be checked,may point to another unknown buffer
 //ptr = wrlim = buffer;
 rdlim = buffer + data.GetCount();
 Seek(npos);
}
void StringStream::Remove(int num,int at){
 if(at==-1)at=GetPos();
 ASSERT(at+num<=GetSize());
 SetReadMode(); //justToKnowWhichToUse...
```

```
 int64 npos=at<=GetPos()?GetPos()-num/*+GetPos()-at*/:GetPos();
 data.Remove(at,num);
 //memmove(buffer+at,buffer+at+num,n);
 buffer = (byte *) ~data;
 rdlim = buffer + data.GetCount();
 Seek(npos);
}
```

Subject: Re: SetLineKeyValue - in memory: minorBugFix
Posted by luoganda on Mon, 01 May 2017 18:55:14 GMT
View Forum Message <> Reply to Message

prevBack sometimes returned empty string - even when there was a value,
chk original message
