
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.Getlter(fpos-1)='='; memcpy((void*)lines.Getlter(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 little - 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.Getlter(fpos)='='; memcpy((void*)lines.Getlter(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(e1,"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;
}

```