Subject: PROPOSAL: small / usefull Stream iface extension
Posted by kohait00 on Wed, 11 Aug 2010 21:39:23 GMT
View Forum Message <> Reply to Message

hi all,

when implementing own, nontrivial protocols using stream, one sometimes (like i am now) in trouble, needing to access the underlying buffer directly, without always triggering a complete copy.. and also beeing able to skip read/write things.
thus here some very small but usefull functions. it would be really cool to have them there.
otherwise please comment why not and how could i circumvent this need, without copying over and over.. (imagine having to put packets there that have some data, over which to calculate crc..etc).

Stream.h:85

```
byte *    Base()              { return buffer; }
byte *    Head()              { return ptr; }
void      SkipRead(dword size = 1) { ptr += min((dword)(uintptr_t)(rdlim - ptr), size); }
void      SkipWrite(dword size = 1){ ptr += min((dword)(uintptr_t)(wrlim - ptr), size); }
```

Head is the most important for me..
SkipRead makes sense as well (ignoring stuff, without need of dummy-copying just to advance ptr)
SkipWrite is kind of just for symetrical completeness, but might be usefull somewhere..

Subject: Re: PROPOSAL: small / usefull Stream iface extension
Posted by kohait00 on Wed, 11 Aug 2010 22:14:38 GMT
View Forum Message <> Reply to Message

and some 2 more..

```
void      PutReserve(byte *& p, dword size = 1) { if(ptr + size <= wrlim) { p = ptr; ptr += size; } else p = NULL; }
dword     GetReserve(byte *& p, dword size = 1) { if(ptr + size <= rdlim) { p = ptr; ptr += size; return size; }
          else { dword s = dword((uintptr_t)(rdlim - ptr)); return ((s>0)?(p=ptr,s):(0)); } }
```

could maybe be named PutShadow, GetShadow, or something, to indicate that the buffer is refered only, no memcpy..
this is usefull to modify the data underneeth directly.

Subject: Re: PROPOSAL: small / usefull Stream iface extension
Posted by kohait00 on Thu, 12 Aug 2010 06:36:15 GMT
View Forum Message <> Reply to Message

the use case of this is the following:

when coding or especially decoding some nontrivial protocols from a Stream (beeing serialized by another device actually, coming in serially) one needs to 'interpret' the data being made available in the stream, instead of consuming it. only if the paket is 'valid' in total, it may be consumed. if not, one may drop one byte and start 'interprete' again. if not all data is available yet one may postpone processing..without copying over and over again..

all this is not possible without accessing the data directly. otherwise a rebuffering would be nesseccary, which is not very well.

trying to avoid copying / reallocating stuff..(will probably run on an embedded system, whithout MPU / MMU (blackfin), which doesnt like a lot of new / delete)

Subject: Re: PROPOSAL: small / usefull Stream iface extension
Posted by mirek on Fri, 13 Aug 2010 07:57:22 GMT
View Forum Message <> Reply to Message

Sorry, but this is poorly defined...

IMO, you should trust more to Seek. I believe that if you reimplement your code with Seek and regular stream interface, I believe it will be as effective as using proposed methods. Well, maybe just almost as effective..

Well, MAYBE we could add something like:

const byte *Peek(int size)

which would return NULL if size is not available...

In any case, SkipRead and SkipWrite do not offer any advantage over Seek IMO.

Subject: Re: PROPOSAL: small / usefull Stream iface extension
Posted by kohait00 on Fri, 13 Aug 2010 08:07:31 GMT
View Forum Message <> Reply to Message

Quote:
Sorry, but this is poorly defined...

thats why i'm asking people who know better

would Peek return the underlying buffer? at ptr position?
then i'm fine with it and can tweak around with Seek ofcorse. indeed, Peek is quite a usefull function, anyway. in fact i prefer it, it keeps the API understandable, not too implementation bound, but usage bound. good idea.

SkipRead / SkipWrite are only helpers, and are not that much nessesary actually, neither my second post. one could would workaround it sth like

Seek(GetPos() + count);

Peek would need to deferentiate wrlim and rdlim right?
why const btw?

const byte * Peek(int size);

my point is solely driven by avoiding unwanted memcpy, thats why accessing buffer directly to perform tweaked read /write (some special communication protocol)

Subject: Re: PROPOSAL: small / usefull Stream iface extension
Posted by kohait00 on Fri, 13 Aug 2010 08:23:55 GMT
View Forum Message <> Reply to Message

another proposal, maybe this one's better...

```
 const byte * Peek(int size = 1)  { if(size <= 0) return NULL; if(ptr + size <= rdlim) return ptr; return NULL; }
 byte *    Head(int size = 1)     { if(size <= 0) return NULL; if(ptr + size <= wrlim) return ptr; return NULL; }
```

Subject: Re: PROPOSAL: small / usefull Stream iface extension
Posted by mirek on Fri, 13 Aug 2010 08:29:07 GMT
View Forum Message <> Reply to Message

kohait00 wrote on Fri, 13 August 2010 04:07Quote:
Sorry, but this is poorly defined...

thats why i'm asking people who know better

would Peek return the underlying buffer? at ptr position?
then i'm fine with it and can tweak around with Seek ofcorse. indeed, Peek is quite a usefull function, anyway. in fact i prefer it, it keeps the API understandable, not too implementation

bound, but usage bound. good idea.

SkipRead / SkipWrite are only helpers, and are not that much nessesary actually, neither my second post. one could would workaround it sth like

Seek(GetPos() + count);

Which is in the interface as SeekCur.

Quote:
Peek would need to deferentiate wrlim and rdlim right?
why const btw?

const byte * Peek(int size);

Sure, only read variant.

Subject: Re: PROPOSAL: small / usefull Stream iface extension
Posted by mirek on Fri, 13 Aug 2010 08:30:07 GMT
View Forum Message <> Reply to Message

kohait00 wrote on Fri, 13 August 2010 04:23another proposal, maybe this one's better...

```
 const byte * Peek(int size = 1)  { if(size <= 0) return NULL; if(ptr + size <= rdlim) return ptr; return NULL; }
 byte *    Head(int size = 1)     { if(size <= 0) return NULL; if(ptr + size <= wrlim) return ptr; return NULL; }
```

For some reasons, write variant sounds somewhat dangerous to me...

Have to think about this one...

Well, in any case, it should advance the ptr...

Subject: Re: PROPOSAL: small / usefull Stream iface extension
Posted by mirek on Fri, 13 Aug 2010 08:46:53 GMT
View Forum Message <> Reply to Message

OK, here is what we got:

```
 const byte *Peek(int size = 1)   { ASSERT(size > 0); return ptr + size <= rdlim ? ptr : NULL; }
 byte       *PutPtr(int size = 1) { ASSERT(size > 0); if(ptr + size <= wrlim) { byte *p = ptr; ptr += size; return p; }; return NULL; }
```

Subject: Re: PROPOSAL: small / usefull Stream iface extension
Posted by kohait00 on Fri, 13 Aug 2010 08:47:20 GMT
View Forum Message <> Reply to Message

another one...

```
 const byte * Peek(int size = 1)  { if(size <= 0 || (ptr + size > rdlim)) return NULL; return ptr; }
 byte *     PutReserve(int size)   { if(size <= 0 || (ptr + size > wrlim)) return NULL; byte * p = ptr; ptr+= size; return p; }
```

concerning Head():
if anyone really wants to break things its even possible with Peek then (brutal cast away const).
those who really use these options are aware of it and use it with caution.

Peek shouldnt advance ptr. Peek only peeks advancing if needed should be done with Seek(GetPos() + size) afterwards, if desired.

advancing ptr in Head is a protection though, that same buffer section is not returned twice.. but the name is maybe irritating.

Subject: Re: PROPOSAL: small / usefull Stream iface extension
Posted by kohait00 on Fri, 13 Aug 2010 08:50:23 GMT
View Forum Message <> Reply to Message

this was work in paralell
i like your option..

my above version were buggy btw p+=size...
