1 General User and Developer Info

1.1 Some notes

- `https://trac.mi.fu-berlin.de/seqan/wiki/IoRevision` is partly outdated, do not rely on it!
- Currently all new IO-Code lives in SeqAn *beside* the old code, you do not need any `#define`s to switch to the new code
- Currently only FastA and FastQ are supported in the new code base
- If you use only FastA and/or FastQ, you should definitely switch now, since the new code is faster and more reliable
- switching is fairly easy, you only need to construct a `RecordReader` object for reading and work on that (see below and dddoc for more details)
- the new calls are `readRecord()`, `read2()` (read entire document), `writeRecord()` and `write2()` (write entire document).

1.2 Some examples

Example code for reading a single record from a gzipped file (you will probably want to loop over `readRecord()`):

```
char filename[] = "/path/to/file.fasta.gz";
gzFile file = gzopen(filename, "rb");
if (gzdirect(file))
{
        // error handling
}
Stream<GZFile> stream(file);

RecordReader<Stream<GZFile>, SinglePass<> > reader(stream);

CharString meta;
DnaString seq;

res = readRecord(meta, seq, reader, Fasta());
if (res != 0)
{
        // error handling
}
```

```
gzclose(file);
```

Example code for reading an entire file efficiently (using memory-mapped strings as files and ConcatDirect-Strings as storage):

```
typedef String<char, MMap<> > TMMapString;
TMMapString stream;
char filename[] = "/path/to/file.fasta";

int res = open(stream, filename, OPEN_RDONLY));
if (res != 0)
{
        // error handling
}

RecordReader<TMMapString, DoublePass<Mapped> > reader(stream);

StringSet<CharString, Owner<ConcatDirect<> > > sequenceIds;
StringSet<DnaString, Owner<ConcatDirect<> > > sequences;

res = read2(sequenceIds, sequences, reader, Fasta());
if (res != 0)
{
        // error handling
}

close(stream);
```

2 Overview of new Stream Module

There is a new Stream module, which currently resides in extras. The module is composed of these main components:

- Stream related (basic abstraction for IO)
  - the stream concept [`concept_stream.h`]
  - Stream adaptations [`adapt_fstream.h`, `adapt_cstdio.h` ..]
  - Stream class and specializations [`stream_*.h`]
- reading related
  - core RecordReader classes (generic Buffer around Stream) [`record_reader*.h`]

  - tokenizing code (low level readUntil* , readWhile* calls etc) [`tokenizing.h is.h`]
  - Format related reading based on tokenizing and RecordReader [`read*.h`]
  - File format detection based on above code [`guess_stream_format.h`]
- Format related writing code [`write*.h`]
- code for casting from string to numerical types [`lexical_cast.h`]

3 Notes on completing the IO-Revision

3.1 General Information

- different Stream backends are not yet implemented (e.g. char array, iostream)
- `tokenizing.h` contains copies of all old parsing/tokenizing functions, please have a look there when switching to new IO (especially since in the old IO everyone confused 'blank' and 'whitespace', as well as other terms!). Please also add any tokenizing functions to `tokenizing.h` and not into your file format code. Do also check if you can use the helper-functions in `tokenizing.h` for your task!
- once all code is ported read2() and write2() shall be renamed to read() and write()

Throughout SeqAn, IO-related code is marked with `//IOREV` and zero or more of the following tags:

Tag	Description
`_nottested_`	This code may not work at all
`_docnottested_`	This code is documented but may behave different from doc
`_nodoc_`	This code should be documented but isn't
`_duplicate_`	This code reimplements functionality found elsewhere
`_delcandidate_`	This code should probably just be deleted
`_notIO_`	This code is falsely tagged as IO
`_noop_`	This call doesnt do anything
`_stub_`	This looks like its not done yet

I would recommend first following the advice in this document and **after** having implemented/ported the relevant code portions, to grep through seqan for IOREV and deal with the leftover code on an individual basis. Since tagging happened in the beginning of my work, some code comments might not by applicable anymore.

3.2 Open Issues in existing files

File	TODOs
`tokenizing.h`	specialized comparison code for other alphabets beside Dna and Dna5
`read_fasta_fastq.h`	Questions what kind of format behaviour is legal (empty sequences? empty headers?)

3.3 Sequence IO

The following previously supported file formats are still missing: embl, genbank, QSeq, raw, CGViz (write-only)

Once they are implemented and all applications have changed to new IO, the following files will be obsolete:

1. `seqan/file/cstream.h`
2. `seqan/file/stream.h`
3. `seqan/file/file_base.h`
4. `seqan/file/file_cstyle.h`
5. `seqan/file/file_filereaderiterator.h`
6. `seqan/file/file_filereader.h`
7. `seqan/file/file_format.h`
8. `seqan/file/stream_algorithms.h`
9. `seqan/file/file_format_raw.h`
10. `seqan/file/file_format_fasta.h`
11. `seqan/file/file_format_embl.h`
12. `seqan/file/file_format_genbank.h`
13. `seqan/file/file_format_cgviz.h`
14. `seqan/file/file_format_mmap.h`
15. `seqan/sequence/sequence_stream.h`
16. `seqan/misc/misc_parsing.h`

The following files shall not be removed, it needs to be decided where to put them in the future:

1. `seqan/file/chunk_collector.h`
2. `seqan/file/file_page.h`
3. `seqan/file/file_page_raid0.h`

4. seqan/file/string_external.h
5. seqan/file/string_mmap.h
6. seqan/file/file_array.h
7. seqan/system/file_directory.h
8. seqan/system/file_async.h
9. seqan/system/file_sync.h

3.4 Store-IO

It was decided that store-io should be ported and moved to the new stream module, this covers seqan/store/store_io*. Other than the beginning of a new sam-implementation (seqan/stream/read_sam.h) none of this has yet happened.

3.5 Alignment and Alignment Graph formats

Includes TCoffeLib, BlastLib, MummerLib, NewickFormat (all in seqan/graph/graph_align_tcoffee_io.h) and fasta-alignment format (in seqan/graph/graph_align_tcoffee_io.h AND in seqan/file/file_format_fasta_align.h).

These should probably be ported as well, but it was not yet decided whether they should be moved to the stream-module or continue to reside in the graph-module.