GBrowse
Generic genome browser

Chen lab workshop

Christian Frech

January 18, 2010

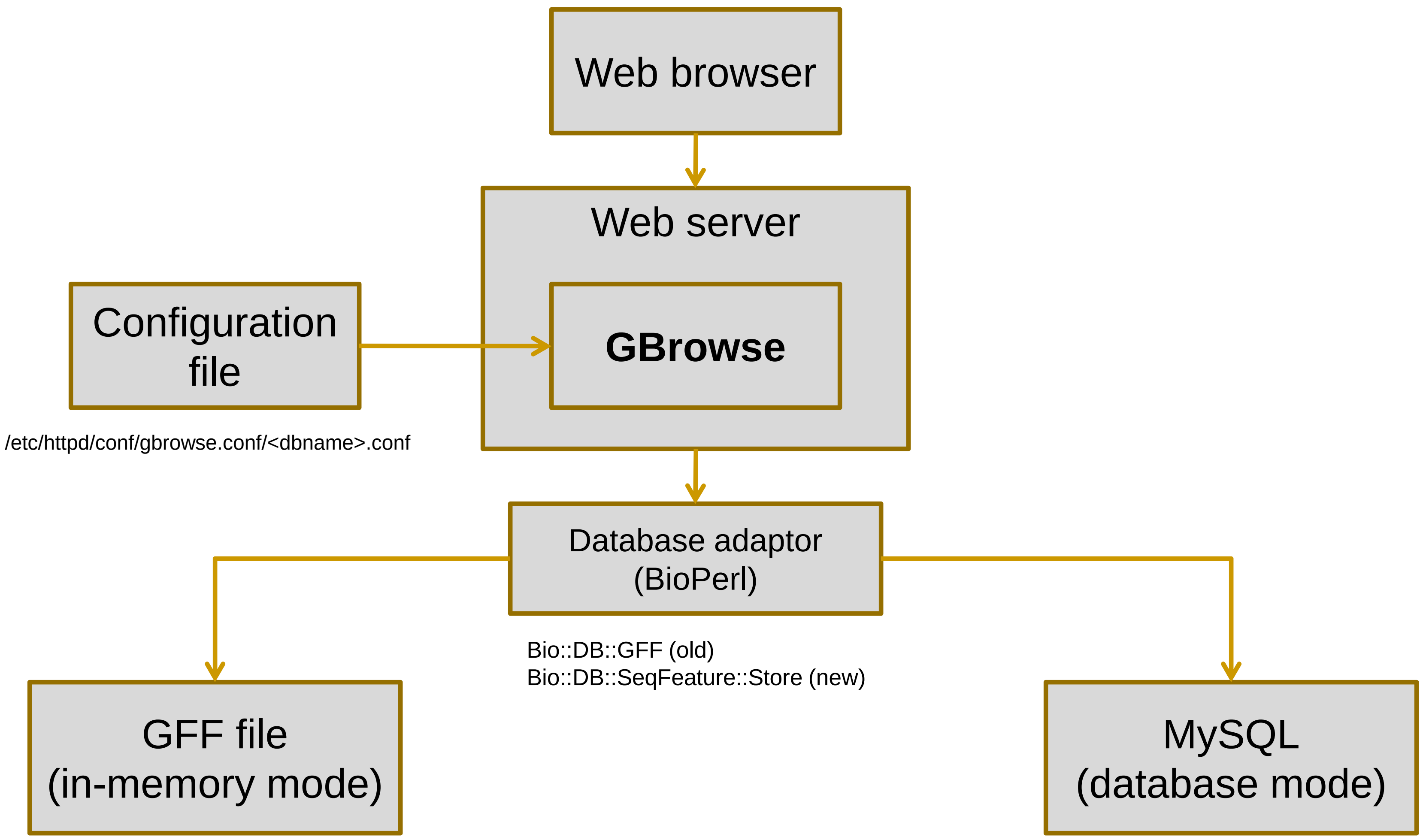
A generic genome browser – why do we need it?

- Genome databases have similar requirements
    - View DNA sequence and its associated features
    - Allow zooming/scrolling
- Do not reinvent the wheel for every genome sequenced

GBrowse facts

- Developed by and most popular component of the Generic Model Organism Database (GMOD) project
  - Collection of open source software tools
- Web-server application implemented in Perl
  - Runs on any machine that runs Perl
- Used by many model organism databases
  - WormBase, FlyBase, PlasmoDB, etc.

GBrowse architecture


Web browser
Web server
GBrowse
Configuration file
/etc/httpd/conf/gbrowse.conf/<dbname>.conf
Database adaptor (BioPerl)
Bio::DB::GFF (old)
Bio::DB::SeqFeature::Store (new)
GFF file (in-memory mode)
MySQL (database mode)

Gbrowse admin tutorial

- This workshop follows the Gbrowse admin tutorial from Lincoln Stein, which can be found at:
  - http://gmod.svn.sourceforge.net/viewvc/gmod/Generic-Genome-Browser/branches/stable/docs/tutorial/tutorial.html?content-type=text%2Fhtml
- Designed for GBrowse 1.69
- Uses GFF3 format files
- Older tutorial based on GFF2 can be found at
  - http://gmod.svn.sourceforge.net/viewvc/gmod/Generic-Genome-Browser/branches/stable/docs/tutorial/dbgff/tutorial.html

Setting up a GBrowse database

- Create a GBrowse directory
  - /var/www/html/gbrowse/databases/<*dbname*>

- Provide configuration file with same name
  - /etc/httpd/conf/gbrowse.conf/<*dbname*>.conf

Running GBrowse off files

- Copy GFF file (and optional FASTA files) to GBrowse database directory
  - /var/www/html/gbrowse/databases/<dbname>/<whatever>.gff
  - /var/www/html/gbrowse/databases/<dbname>/<whatever>.fa
- Set directory where GFF files are located in configuration file

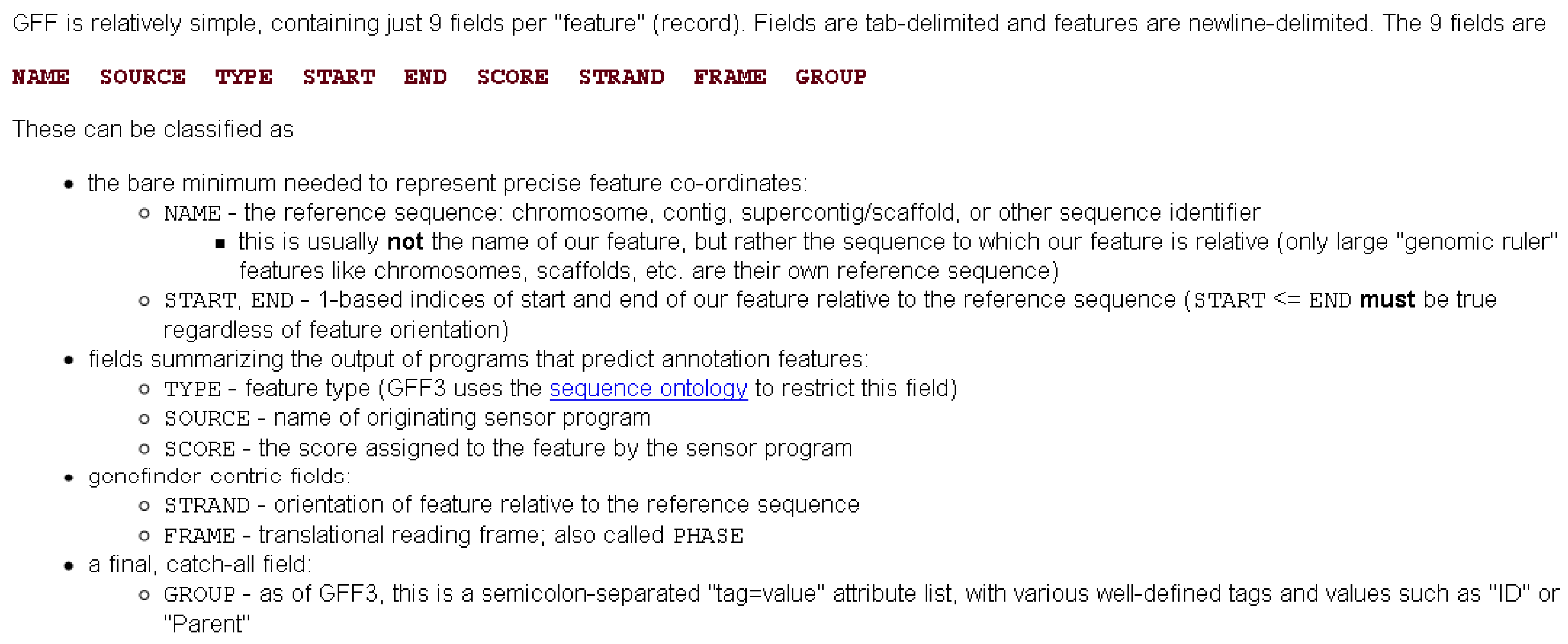
  -
    ```
    [GENERAL]
    description   = Volvox Example Database
    db_adaptor    = Bio::DB::GFF
    db_args       = -adaptor memory
                    -gff /var/www/html/gbrowse/databases/<dbname>
    ```

Configuration file

- /etc/httpd/conf/gbrowse.conf/<*dbname*>.conf
- Some things you can (or have to) do with it:
  - Set database adaptor
  - Configure the GBrowse web page (selectable zoom factors, example regions, etc.)
  - **Define feature tracks**
    - Data source
    - How they are displayed (glyphs)

General feature format (GFF)

- Widely used file format describing genomic features
- Current version GFF3; GFF2 is depricated!



GFF is relatively simple, containing just 9 fields per "feature" (record). Fields are tab-delimited and features are newline-delimited. The 9 fields are

NAME SOURCE TYPE START END SCORE STRAND FRAME GROUP

These can be classified as

- the bare minimum needed to represent precise feature co-ordinates:
  - `NAME` - the reference sequence: chromosome, contig, supercontig/scaffold, or other sequence identifier
    - this is usually **not** the name of our feature, but rather the sequence to which our feature is relative (only large "genomic ruler" features like chromosomes, scaffolds, etc. are their own reference sequence)
  - `START`, `END` - 1-based indices of start and end of our feature relative to the reference sequence (`START` <= `END` **must** be true regardless of feature orientation)
- fields summarizing the output of programs that predict annotation features:
  - `TYPE` - feature type (GFF3 uses the sequence ontology to restrict this field)
  - `SOURCE` - name of originating sensor program
  - `SCORE` - the score assigned to the feature by the sensor program
- genefinder centric fields:
  - `STRAND` - orientation of feature relative to the reference sequence
  - `FRAME` - translational reading frame; also called `PHASE`
- a final, catch-all field:
  - `GROUP` - as of GFF3, this is a semicolon-separated "tag=value" attribute list, with various well-defined tags and values such as "ID" or "Parent"

Example GFF3

```
ctgA example gene            1050 9000 . + . ID=EDEN;Name=EDEN;Note=protein kinase

ctgA example mRNA            1050 9000 . + . ID=EDEN.1;Parent=EDEN;Name=EDEN.1;Index=1
ctgA example five_prime_UTR  1050 1200 . + . Parent=EDEN.1
ctgA example CDS             1201 1500 . + 0 Parent=EDEN.1
ctgA example CDS             3000 3902 . + 0 Parent=EDEN.1
ctgA example CDS             5000 5500 . + 0 Parent=EDEN.1
ctgA example CDS             7000 7608 . + 0 Parent=EDEN.1
ctgA example three_prime_UTR 7609 9000 . + . Parent=EDEN.1

ctgA example mRNA            1050 9000 . + . ID=EDEN.2;Parent=EDEN;Name=EDEN.2;Index=1
ctgA example five_prime_UTR  1050 1200 . + . Parent=EDEN.2
ctgA example CDS             1201 1500 . + 0 Parent=EDEN.2
ctgA example CDS             5000 5500 . + 0 Parent=EDEN.2
ctgA example CDS             7000 7608 . + 0 Parent=EDEN.2
ctgA example three_prime_UTR 7609 9000 . + . Parent=EDEN.2

ctgA example mRNA            1300 9000 . + . ID=EDEN.3;Parent=EDEN;Name=EDEN.3;Index=1
ctgA example five_prime_UTR  1300 1500 . + . Parent=EDEN.3
ctgA example five_prime_UTR  3000 3300 . + . Parent=EDEN.3
ctgA example CDS             3301 3902 . + 0 Parent=EDEN.3
ctgA example CDS             5000 5500 . + 1 Parent=EDEN.3
ctgA example CDS             7000 7600 . + 1 Parent=EDEN.3
ctgA example three_prime_UTR 7601 9000 . + . Parent=EDEN.3
```

```
[Genes]
feature             = gene
glyph               = gene
bgcolor             = peachpuff
label_transcripts   = 1
draw_translation    = 1
category            = Genes
key                 = Protein-coding genes
```

Example GFF2

```
IV     curated  mRNA   5506800 5508917 . + .   Transcript B0273.1; Note "Zn-Finger"
IV     curated  5'UTR  5506800 5508999 . + .   Transcript B0273.1
IV     curated  exon   5506900 5506996 . + .   Transcript B0273.1
IV     curated  exon   5506026 5506382 . + .   Transcript B0273.1
IV     curated  exon   5506558 5506660 . + .   Transcript B0273.1
IV     curated  exon   5506738 5506852 . + .   Transcript B0273.1
IV     curated  3'UTR  5506852 5508917 . + .   Transcript B0273.1
```

- No more than two hierarchy levels
  - parent and childs
- Grouping of features by assigning identical tags values
- Aggregators required to display features with complex structure
  - Need to be configured in .conf file (ask Jeff or Ismael ;-)

Running GBrowse off MySQL database

- Set up MySQL database
  - Ask Duncan or me :-)
- Upload GFF file into database
  - bp_load_gff.pl – incremental loading into existing database; slow
  - bp_bulk_load_gff.pl – initialize DB from scratch; 10x faster; **deletes database**!
  - bp_fast_load_gff.pl – incremental loading as fast as bulk load; does not work on all platforms
  - bp_seqfeature_load.pl – new (fast) load script fully compatible with GFF3
- Change database adaptor in .conf file

```
[GENERAL]
description   = Volvox Example Database
db_adaptor    = Bio::DB::SeqFeature::Store
db_args       = -adaptor DBI::mysql
                -dsn     volvox
                -user    nobody
```

GBrowse 2.0 – the next generation

- Rewrite of the original Gbrowse
  - Currently in alpha test stage
- Features
  - Dynamic updating via AJAX (“smooth scrolling”)
  - Attach different databases to GBrowse tracks
  - Render different tracks in parallel (on different machines) to significantly increase performance
- http://modencode.oicr.on.ca/cgi-bin/gb2/**gbrowse**/worm/

Useful links

- GBrowse
  - http://gmod.org/wiki/GBrowse
- Gbrowse mailing list
  - http://sourceforge.net/mailarchive/forum.php?forum_id=31947
- GFF3 specification
  - http://song.sourceforge.net/gff3.shtml
- GFF3 validator
  - http://modencode.oicr.on.ca/cgi-bin/validate_gff3_online
- BioPerl documentation
  - http://doc.bioperl.org/
- Glyphs available in GBrowse
  - http://bioperl.org/wiki/Module:Bio::Graphics::Glyph

Using BioPerl to programmatically retrieve data from GFF3 files (or from database)

```
#!/usr/bin/perl

use strict;
use warnings;
use Bio::Perl;
use Bio::DB::SeqFeature::Store;
use Bio::DB::GFF;

# get feature from GFF file
my $db = Bio::DB::SeqFeature::Store->new
(
        -adaptor => "memory",
        -dir => "/var/www/html/gbrowse/databases/workshop_gff3“
);
my ($gene) = $db->features(-name => "EDEN.1", -aliases => 1);
```

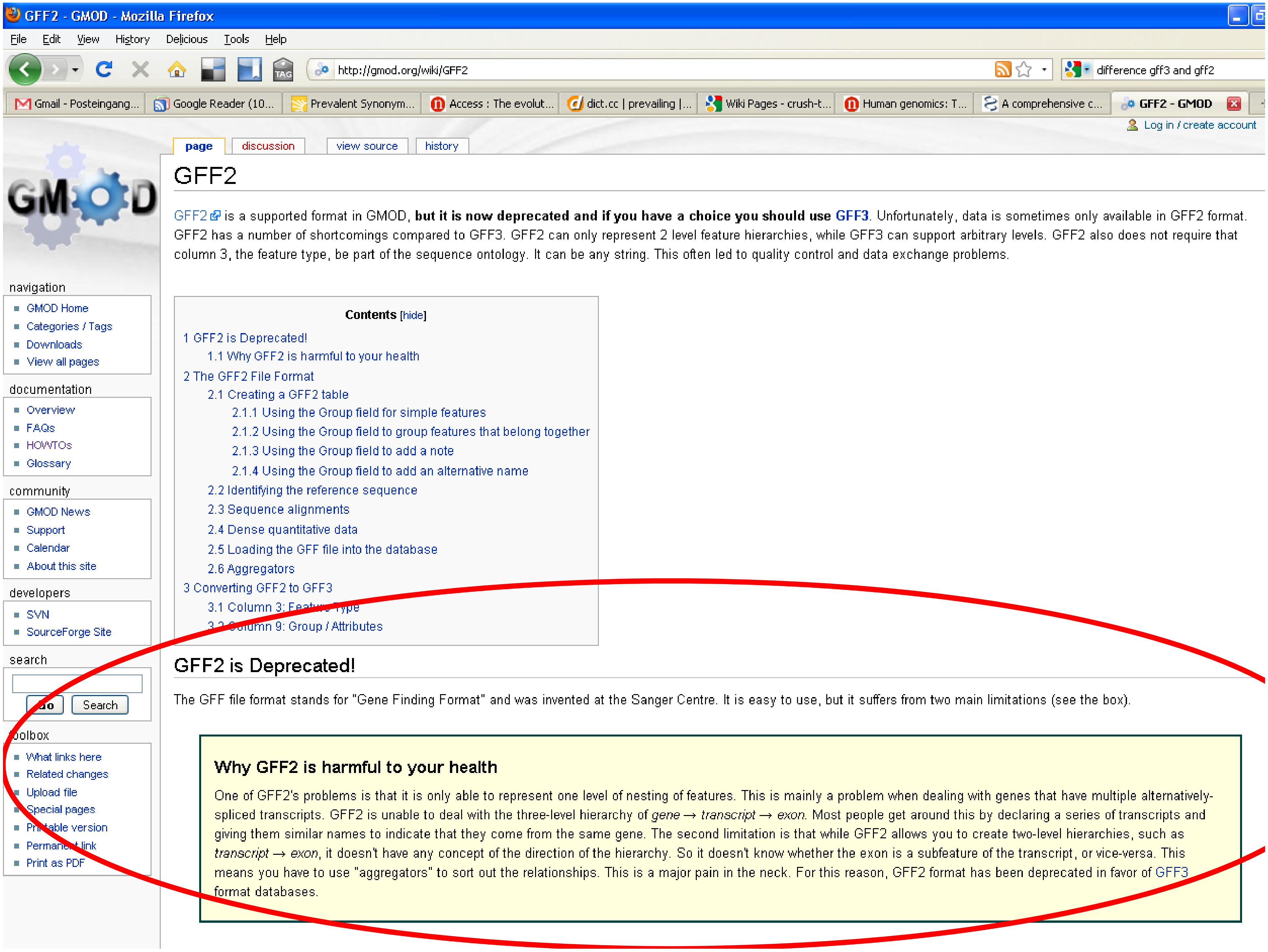
```
print "Feature type: ".ref($gene)."\n";
print "Feature name: ".$gene->display_name."\n";
print "Feature start coordinate: ".$gene->start."\n";
print "Feature end coordinate: ".$gene->end."\n";
print "Feature strand: ".$gene->strand."\n";

# get coding sequence of transcript;
# this code works only if gene is on forward strand
my $cds = "";
$cds .= $_->dna foreach sort {$a->start <=> $b->start} $gene->CDS;
print ">CDS\n$cds\n";

# get protein sequence of transcript
print ">Protein sequence\n".translate($cds)->seq."\n";
```

Thank you

Christian Frech


GFF2 - GMOD - Mozilla Firefox

File Edit View History Delicious Tools Help

http://gmod.org/wiki/GFF2

difference gff3 and gff2

Gmail - Posteingang... | Google Reader (10... | Prevalent Synonym... | Access : The evolut... | dict.cc | prevailing |... | Wiki Pages - crush-t... | Human genomics: T... | A comprehensive c... | GFF2 - GMOD

Log in / create account

page | discussion | view source | history

GFF2

GFF2 is a supported format in GMOD, **but it is now deprecated and if you have a choice you should use GFF3**. Unfortunately, data is sometimes only available in GFF2 format. GFF2 has a number of shortcomings compared to GFF3. GFF2 can only represent 2 level feature hierarchies, while GFF3 can support arbitrary levels. GFF2 also does not require that column 3, the feature type, be part of the sequence ontology. It can be any string. This often led to quality control and data exchange problems.

Contents [hide]

navigation
- GMOD Home
- Categories / Tags
- Downloads
- View all pages

documentation
- Overview
- FAQs
- HOWTOs
- Glossary

community
- GMOD News
- Support
- Calendar
- About this site

developers
- SVN
- SourceForge Site

search

Go | Search

toolbox
- What links here
- Related changes
- Upload file
- Special pages
- Printable version
- Permanent link
- Print as PDF

GFF2 is Deprecated!

The GFF file format stands for "Gene Finding Format" and was invented at the Sanger Centre. It is easy to use, but it suffers from two main limitations (see the box).

Why GFF2 is harmful to your health

One of GFF2's problems is that it is only able to represent one level of nesting of features. This is mainly a problem when dealing with genes that have multiple alternatively-spliced transcripts. GFF2 is unable to deal with the three-level hierarchy of *gene* → *transcript* → *exon*. Most people get around this by declaring a series of transcripts and giving them similar names to indicate that they come from the same gene. The second limitation is that while GFF2 allows you to create two-level hierarchies, such as *transcript* → *exon*, it doesn't have any concept of the direction of the hierarchy. So it doesn't know whether the exon is a subfeature of the transcript, or vice-versa. This means you have to use "aggregators" to sort out the relationships. This is a major pain in the neck. For this reason, GFF2 format has been deprecated in favor of GFF3 format databases.