```
                              BIBLIO
                     bibliographic preprocessor


                           09 April 2001
                           27 June  2001
                           07 Sept  2001 (ver 1.1)
                           17 Sep   2001 (ver 1.11)
                           04 Oct   2001 (ver 1.2)
                           12 March 2002 (ver 1.21)
```

CONTENT

==

Abstract

Biblio is a programm for preprocessing your bibliographic references. It is written in Perl and does not use the LaTeX internal language. After preprocessing the input TeX file it creates output TeX file with references included according to the formats prescribed either in a separate file (called "template" or "format file" further on)
or in the header of the input TeX file. Output TeX is ready for LaTeX.
If you do not need details, move to the section 8 (and 9) of this text.

----

0. COPYRIGHT

----

1. STATEMENT OF PURPOSE

Don't reinvent a wheel - this should be your first reaction. Might be you are right: many people use BibTeX or some other (I don't know which but pretty sure they do exist) programs to pre-process their bibliographic references lists. But what I did before was just including all citations directly into my TeX-file. Before... Before this story has happened.

I was writing an article and from the very beginning we (I and my bosses) decided to send it to the Journal A. Of course, I wrote everything including

citations in the style of this journal. But when the work was almost finished we (you know whom I mean) decided to send the paper to another journal, say Journal B. And of course it has different rules and standards.
No problem - one should say,- Use BibTeX. Yes, that's OK - is my reply.
But what I basicly needed was to change the order of names (last, first, etc) in the authors list. Like instead of "J.J.Smith" now I needed "Smith, J.J.".
Might be information I used was quite outdated, but I could not find the way to handle this kind of things with the BibTeX.

Another reason to start this project was handle somehow ordering of the citation list. Normally articles in the "References and Notes" section should appear in the same order as they are in the body-text. But sometimes it's quite useful to have the list ordered according to the first author name or something different.

And finally, some journals prefer to have "as raw text as possible". They do accept TeX-files but... without cross references and citations which means, that everything have to be done by hands and ONLY ONE file has to be submitted. I don't like this idea! But I DO want to be published there! So, may be a partial solution is (for citations at least) is to pre-process the TeX-file to another TeX-file where all \cite and \bibitem commands are substituted by normal text. Of course, according to some pattern. This solves one more problem - I'm not very good in TeX programming and find it quite difficult to make TeX command \cite produce some nice output in the form ^{1,2,6} instead of [1,2,6].

So, let's move to the GENERAL IDEAS

2.GENERAL IDEAS

  A. This program pre-processes TeX-file making another TeX-file.

  B. Citations are inserted from the DataBase(s) which has form of a separate file. Fields of the records in this file are absolutely ARBITRARY, except for LABEL and TYPE. Field AUTHOR has also some special meaning which will be clarified further on.
This DataBase has form of

```
======================================================
LABEL   = Landau
TYPE    = book
AUTHORL = Landau|Lifshits
AUTHORF = L     |E.
AUTHORM = D.    |M.
TITLE   = Statistical Physics
PUBL    = Pergamon
PLACE   = Oxford
YEAR    = 1980
----------
LABEL   = Vasilevskaya
TYPE    = article0
AUTHORL = Vasilevskaya|Klochkov|Khalatur|Khokhlov|Brinke
AUTHORF = V.          |A.      |P.      |A.      |G.
AUTHORM = V.          |A.      |G.      |R.      |
TITLE   = Microphase separation within comb-like copolymer with attractive  side
-chains: computer simulations.
JOURNAL = Macromol. Theory and Simul.
STATE   = accepted
----------
LABEL   = Helfand1
TYPE    = article
```

```
AUTHORL = Helfand|Wasserman
AUTHORF = E.     |Z.
AUTHORM =        |R.
JOURNAL = Macromolecules
VOL     = 9
PAGE    = 879
YEAR    = 1976
----------
LABEL   = 3dFlex
TYPE    = article
AUTHORP =         |        |      |ten
AUTHORL = Subbotin|Saariaho|Ikkala|Brinke
AUTHORF = A.      |M.      |O.    |G.
TITLE   = Elasticity of comb copolymer cylindrical brushes
JOURNAL = Macromolecules
PAGE    = 3447
YEAR    = 2000
VOL     = 33
----------
LABEL   = Dobrynin1
TYPE    = article
AUTHORL = Dobrynin|Erukhimovich
AUTHORF = A.|I.
AUTHORM = V.|Y.
TITLE   = Computer-aided comparative investigation of architecture influence of
block copolymer phase diagrams
JOURNAL = Macromolecules
PAGE    = 276
EPAGE   = 281
YEAR    = 1993
VOL     = 26
COMMENT = Read it before you have graduated!!!
----------
====================================================================

Please, note that the last entry has EPAGE (which means "end page") and title
also included whereas record "Helfand1" contains much less information.
This will not cause any problem. The only person to whom you have to agree is
yourself: if you call the first (starting) page of the paper (book, etc) "PAGE",
do it for all entries. If you prefer to call it "EGAP" - it's your choice!

One can use abbreviation in the form of TeX commands, like
"\mml" can mean "Macromol." if we include line "\def\mml{Macromol.}"
somewhere in the tex-source. To do so see "-a" option: to point to the
abbreviations file "-a abbr.tex" should be used. Note: Biblio will not check
consistency of your abbreviations!!!

LABEL and TYPE are predefined! Also AUTHOR is predefined. "AUTHOR" without "L",
"R" or anything else at the end. It is predefined in the sense, that entries in
the database, which start with "AUTHOR" ("AUTHORL", "AUTHORF") are considered to
be arrays with separator "|". There are two equivalent ways to represent them
in the DataBase:
* one is shown above (with "|" as separator)
* the "Dobrynin1" entry can be also written in the form
==============
AUTHORL = Dobrynin
AUTHORF = A.
AUTHORM = V.
AUTHORL = Erukhimovich
AUTHORF = I.
AUTHORM = Y.
===============
```



```
The only important issue is the order of this entries: the first AUTHORF belongs
to the first AUTHORL. In principle you can include anything between them, like:
===============
AUTHORL = Dobrynin
YEAR    = 1993
AUTHORF = A.
AUTHORM = V.
PAGE    = 276
AUTHORL = Erukhimovich
AUTHORF = I.
AUTHORM = Y.
==============
It will produce the same result.

If there are several entries for the same key (like YEAR) which is NOT supposed
to be array (i.e. does not start with AUTHOR), then only the last one will
influence the final result.

The fact that all arrays should start with AUTHOR does not mean that you can
apply it to the authors' list only. It will become more transparent further.

 C. Of course you want some nice output. Your LaTeX document should look like:

====
\documentstyle[12pt,a4]{article}
\title{Blah-blah}
\begin{document}\maketitle

Text text \cite{Dobrynin1}.
Text Text text \cite {Dobrynin1,3dFlex}

\begin{thebibliography}

\end{thebibliography}

Something more you need (list of figures or whatever)

\end{document}
====

Note, that NOTHING is between \begin{thebibliography} and
\end{thebibliography}. These two are just signs for Biblio to insert your
references list in between.

What does Biblio do? It reads your TeX-file and finds all references (which you
have in \cite command) in the database. Then it inserts them according to some
formats (see further) between \begin{thebibliography} and \end{thebibliography}.
It can do it in two ways:

FIRST: insert them as "\bibitem{LABEL} TEXT OF THE REF" and not touching
anything else (text of the ref is according to the format, of course);

SECOND: insert them in some special way, where you can define what \cite will
produce and what kind of numbering you want in the references list. For example,
if you like $^{1,5-7}$ more than standard [1,5-7], normally produced by \cite,
and
(1) ref1
(2) ref2
instead of
[1] ref1
[2] ref2
produced by \bibitem.
```

```
(Hint: your choice between FIRST and SECOND you can do by using "--nobibitem"
parameter, and "cite_format" and "ref_format" fields in the format file).

After it's done, you can process output file with normal LaTeX.

(Hint: You can find script "libtex" which will help you to handle it.)

 D. All formats are stored in the format file (see -f command line option)
or just in your TeX document's header. About rules (how to write this formats)
you can read below.


 E. So, your checklist is:
    1. Input TeX file.
    2. References database.
    3. Format file

If you have all these you have to do

biblio  -o outdoc.tex -d refsdata.db refsdata1.db -f formats.db   indoc.tex

where
- outdoc.tex  - output filename
  (if not given, them "biblio."-prefix is added to "indoc.tex"
  like "biblio.indoc.tex")
- refsdata.db - references database (may be more than one)
- formats.db  - format file (only ONE)
- indoc.tex   - input file

The order of the parameters is not important.

-------------------------------------

3. STRUCTURE OF THE BIBLIO-DATABASE

It has form described in 2.B. It consists of records. After each record a
separator has to be placed. By default, it is "----------\n". You can easily
change it by command line option "--separator". Only on separator can be used!

Each record has fields. One line per one field - is a rule!
You cannot change it (without looking into the source, of course)!

Each field has a structure NAME = VALUE.

Fields' names LABEL and TYPE are predefined - each record has to have them.
Otherwise it will be ignored.

Field AUTHOR has a special structure.
Usage is quite obvious from an example

AUTHORP =         |        |      |ten
AUTHORL = Subbotin|Saariaho|Ikkala|Brinke
AUTHORF = A.      |M.      |O.    |G.
AUTHORM = V.

There are 4 authors: Subbotin A.V.; Saariaho M.; Ikkala O. and ten Brinke G.

P, L, F and M are from "prefix", "last", "first", "middle".

Sign "|" separates authors. It is reserved and cannot be changed!
```



```
You can use as many fields as you like. If you don't use them in the
output - no problem, part of them will be ignored.

You can have several databases. Use inline option "-d" to say Biblio names
of databases you want to use.

That's all!

-------------------------------------

4. STRUCTURE OF THE FORMAT FILE (FORMAT STRING)

Your output (formated entries from database put into the TeX document) has to
be formated according to the rules described in the so called format file (or
templates) or just in the beginning of your TeX document.
The first way is preferable - it gives taste of more structural ordering.
Use option "-f" to point to a format file. Otherwise Biblio will try to find
formats in the TeX-file and, if not succeeded, will complain and halt.

What we want to format?
1. References list (what you normally have at the end of your paper)
2. Citations in the text (what you do with \cite)

Before reading this, read 2.C from "General Ideas" section.

How to write formats? That's the hardest part!

Let's start from an
EXAMPLE:
-------------------
% this is template for Macromolecules
% no ordering of the references is needed. Otherwise add line
% '%ordering= AUTHORL' - ordering with key field 'AUTHORL'

%begin{biblio}

%cite_format={(%s)}
%ref_format=${^{%s}}$

%% normal article:
% ARTICLE [$AUTHOR0P~L,~F~M; $ {\it $JOURNAL$}  {\bf $YEAR$}, {\it $VOL$}, $PAGE
$.]

%% article which is submitted or accepted
% ARTICLE0[$AUTHOR0P~L,~F~M; $ {\it $JOURNAL$}  $STATE$.]

%% normal book
% BOOK    [$AUTHOR0P~L,~F~M; $ {\it $TITLE$}; $PUBL$: $PLACE$, $YEAR$.]

%% article in a book AUTHORE, AUTHORG and AUTHORH - last, first
%% and middle names of the editor
% EDBOOK  [$AUTHOR0P~L,~F~M; $ in {\it $BOOKTITLE$}, edited by $AUTHOR0I~E,~G~H;
$; $PUBL$: $PLACE$, $YEAR$.]

%% article in a book without editor
% INBOOK  [$AUTHOR0P~L,~F~M; $ in {\it $BOOKTITLE$}, p.$PAGE$ $PUBL$: $PLACE$, $
YEAR$.]

% EPRINT [$AUTHOR0P~L,~F~M; $ in {\it $BOOKTITLE$} $INFO$, $YEAR$.]

%end{biblio}
-------------------
```

```
4.0. Syntax

  A. Each line starts with %-sign. This is needed because in principle this can
     be included into your TeX-document, so it has to be considered as a comment
     by LaTeX. Lines, starting with %% are "REAL"  comments (i.e. they are
     comments for Biblio too).

  B. All this information has to be stored either in a template file (format
     file) [use option -f filename] or just in the header of your TeX document.
     In both cases it has to start with '%begin{biblio}' line and end with
     '%end{biblio}'.
     Only one template file is allowed (you cannot read from 2 for the
     same document).

  C. There are some OPTIONAL reserved keys. If omitted, default values are
     assumed and warning is printed.
     Reserved keys are
     C.1.   cite_format:  has a form %cite_format={(%s)}
            %s just means "string". {(%s)} will produce output in your
            references list like

            (1) Simpson,....
            (2) Tompson...
            ...

            or %cite_format={%s.} will produce

            1. Simpson,....
            2. Tompson...
            ...


     C.2.   ref_format: has a form  %ref_format=${^{%s}}$
            This means how your \cite command output will look like.
            For example, we have \cite{Aw1,Paper3,John} in the TeX document and
            %ref_format=${^{%s}}$ in your format file. Then you get

            ${^{2,4,5}}$    (numbers depend on what is ordering, of
                             course)
            or
            ${^{2-4}}$   if they all are going one by one in your refslist.

            in the output TeX file instead of your \cite command.
            If you have  %ref_format={[%s]} then

            {[2,4,5]}
            or
            {[2-4]}

     C.3.   ordering: has a form  %ordering= AUTHORL
            This means that the field AUTHORL has to be considered as a key field
            in ordering of your references. You can put whatever you want there,
            like YEAR, VOL,...
            If this filed is not present in some entries, it considered to be
            empty.

  D. Each TYPE (TYPE is article, book or something like this) has to have an
     entry in the format file. This entry (format string) has a form

     % TYPE [format string]

     (it MUST be in ONE line!!!)
```



```
     where the "format string" says to Biblio how to output information about
     your reference of the type TYPE.
     "[" and "]" MUST be there! They say where you start your format line and
     where it ends (if you need to use inside the format line, don't worry:
     the first [ is the start and the last ] is the end  of the string;
     all the rest are just characters).
     Everything between $-signs is a filed name. $VOL$ will be substituted by
     54 if you have an entry
     VOL = 54
     for this given article in your database.
     It's a bit more tricky for AUTHOR (see below).
     All the other words are just normal text or LaTeX commands.
     Example:

     Let say, you have the following item in your database file

     LABEL   = LifshitzGrosbergKhokhlov1
     TYPE    = article
     AUTHORL = Lifshitz |Grosberg |Khokhlov
     AUTHORF = I.       |A.       |A.
     AUTHORM = M.       |Yu.      |R.
     JOURNAL = Rev. Mod. Phys.
     VOL     = 50
     PAGE    = 683
     EPAGE   = 713
     YEAR    = 1978
     TITLE   = Some problems of the statistical physics of...

     and an entry

     % ARTICLE [$AUTHOR0P~L,~F~M; $ {\it $JOURNAL$}  {\bf $YEAR$}, {\it $VOL$},
$PAGE$.]

     in your format file (I assume also '%cite_format={(%s)}').
     Then output will be

     (1) ~Lifshitz,~I.~M.; ~Grosberg,~A.~Yu.; ~Khokhlov,~A.~R.
         {\it {Rev. Mod. Phys.}}  {\bf {1978}}, {\it {50}}, {683}.


4.1. AUTHOR-word:

  A. Word AUTHOR is reserved and assumed to be an array.
     This means, that AUTHORLFM is in fact
     hash index with keys AUTHORL, AUTHORF, AUTHORM,
     or, if you like
     AUTHOR[L]='Smith'
     AUTHOR[F]='J.'
     AUTHOR[M]='A.'

  B. After the word AUTHOR follows the max number of authors to show
     (let us call it max_A).
     If actual number of authors n_A > max_A,
     then "$reservedET_AL" will be written after the
     author number max_A and  names of all the others will be omitted.
     "max_A==0" (zero) means "max_A==infty"
  C. $AUTHOR0L, F; $ means that fields should be written as follows:
         1) "AUTHORL"
         2) ", "
         3) "AUTHORF"
         4) "; "  (let us  denote it as "last_sep")
         5) go to "1)" for the next author
```

```
     For example, for the Lifshitz' article above:

     $AUTHOR0L, F, M; $ produces
     Lifshitz, I. M.; Grosberg, A. Yu.;  Khokhlov, A. R.

     or

     $AUTHOR1L, F; $ produces
     Lifshitz, I. et al

     Note, there is no the middle name in the 2nd example, because there is
     no "M" in the format line.

  D. There is a special sign "|" which means that AUTHOR format line is ended
     in fact and everything after it is so called "and_word".
     This "and_word" (normally, it contains only "and") is something  what is
     written before the last author if the number of authors in larger than 1.

     Example:
     $AUTHOR0L, F, M; |and$ produces
     Lifshitz, I. M.; Grosberg, A. Yu.; and  Khokhlov, A. R.


     Note, that
         1) if "et al" have to be printed, "and_word" will not be there!

            $AUTHOR2L, F; |and$ produces
            Lifshitz, I.; Grosberg, A. Yu. et al

         2) if number of authors is larger than 2 but less or equal to max_A:
            before the "end_word" the "last_sep" has to be written
            otherwise - if only 2 authors: print "and_word", but no "last_sep"

            Example:  Q.W. Rom and R.T. Lai
                 but  Q.W. Rom, T.G. Doi, and R.T. Lai
               here:  last_sep="," ; and_word="and")

                      This example corresponds to
                      $AUTHOR0FM L, |and$


4.2. Fields LABEL, TYPE and AUTHOR are predefined. All the others can be named
   by you. The only condition is: your fields in the database and in the formats
   have to coincide. If some field is present in the format string but there
   is no value of it in the database, sign $reservedEMPTY_FIELD_SIGN will be
   printed out instead (see command line parameter --empty_field_sign);
   by default $reservedEMPTY_FIELD_SIGN == '???'.
   My suggestion is: use YEAR, VOL, PAGE, etc. for the year, volume and page and
   don't name page as RTY45 - this is mush less readable.

-------------------------------------

5. RUNNING BIBLIO

So, now you have references database and format file. It is time to talk about
command line parameters. Here they are:

 ESSENTIAL PARAMETERS

+ -o output.file  => $OUT_TEX ;
```



```
                      if no given, assuming $OUT_TEX = "biblio.$TEX_NAME"
+ -f format.file  =>  $FORMAT_NAME
                      (can introduce templates for Macromolecules,
                      etc. If given, "in-tex" parameters will be
                      ignored!!!)

+ -d dbase.file   =>  @REFS_FILES (where all info about refs is!)
  accepts many arguments

+ -a abbrev.file  =>  file of abbreviations (tex file)
                      if given, will be inserted just BEFORE
                      "\begin{references}" in the output.file.
                      Consists of anything you want, but the idea is
                      to use it like "\def\mml{Macromol.}"
                      and  have entry
                         JOURNAL = \mml
                      instead of
                         JOURNAL = Macromol.
                      in the dbase.file.

+ -v <number>     =>  verbosity level (0-5) $verbosity_level

+ -h or --help    =>  short help information

+ --nobibitem     =>  $optionsKEEP_BIBITEM_AND_CITE=0. Default is 1!!!
                      (read 2.C too)
                      If not given, Biblio will insert all references
(formated, of course) into the out-TeX document, and that's it! You will
have  \cite-commands as you had them before.  List of references will be
produced by LaTeX itself (with \bibitem).
If given (but NO --uselist), it will print your references as a simple text,
flushed left.


+ --uselist       =>  $optionsREFS_AS_TEX_LIST=1 Default is 0!!!
                      Will output the references list as a TeX list
with counter. Looks much better than \flushleft (try it!)


(if no --nobibitem nor --uselist is given it is assumed to use \bibitem
 if --nobibitem is given, it will use \flushleft
 if --nobibitem AND --uselist are given it will use \list
)


 ADDITIONAL PARAMETERS

     key                                variable in the source,default value
                                                     and some info

+ --etal "{\it{et al}}"            =>  form of "et al"-string in $reservedET_AL
                                       (see 4.1.C)
+ --separator "----------\n"       =>  separator in the database-file

+ --not_insert_empty_field_sign    =>  $optionsINSERT_EMPTY_FIELDS_SIGN=0
                                       If some field is present in the format
string for the given TYPE, but is not in the database, Biblio will
insert some sign instead (see --empty_field_sign).

+ --empty_field_sign xxx           =>  $reservedEMTY_FIELD_SIGN = '???'
                                       (see --not_insert_empty_field_sign)
```

```
+ --start_refs                      => @STARTREFS=
  accepts many arguments      ('begin{references}','begin{thebibliography}');
                                    Tells where Biblio has to start inserting
                                    the references list.

+ --end_refs                        => my @ENDREFS=  ....
   accepts many arguments

? --bibitem_string                  => my $BIBITEM = '%'."\n".'\bibitem{';

+ --version                         => version info

+ --movepoint                       => added in ver1.1
                                       make "word.\cite{} Another sent"
                                       from "word \cite{}. Another sent"
      Everything in the variable $what_we_call_point='.,;:' is considered
      to be a "point".
      This way of making citations is required by some journals
      (for instance, Macromolecules)  which have citation in the form
                          "in recent works$^1-5$"
      There point (or comma,...) has to be placed BEFORE citation, like
                 'works.$^{1-5}$' but NOT 'works$^{1-5}$.'
      NOTE: this will work only for point IN THE SAME LINE, like
      'was show recently \cite{q2,lab4,yt6}. Therefore'
      but NOT for
      'was show recently \cite{q2,lab4,yt6}
      . Therefore'

(not everything is accomplished. So, ?-sign means, that in your version
this parameter might be still not working)

-------------------------------------

6. EXAMPLES

   A. I have a couple of format files (in the directory called "formats")
and a database. First, about the structure, I decided to follow. (It does
NOT mean you have to do it in the same way!)

Types I use are

ARTICLE    just regular article
ARTICLE0   submitted or accepted article

BOOK       regular book
EDBOOK     article in a book (with an editor!)
INBOOK     article in a book without editor

EPRINT     electronic preprint

Some examples of format files you can find in the "formats" directory.

Example of a database is called "refers.db".


  B. Document "EXAMPLES/2sorts/2sorts.tex" is an introduction
to one of my articles.  You can use it as an example.
It needs templates which are in the "formats" dir.
Just type "./libtex" and that's it!

  C. Document "EXAMPLES/2sorts/2sorts_formats.tex" is
the same as in B., but all formats are included into the header
```



```
of the document. Read this header to find out how to compile it.

  D. Document "EXAMPLES/hairy/hairy.tex" shows how to use abbreviations.

-------------------------------------

7. ADDITIONAL SCRIPTS

You can find some more scripts in the top directory.

   A. libtex
      is a useful script for processing TeX documents on the
      stage of their development. It can be easily adjusted
      (open it in any text editor and change parameters). The idea is
      to keep its copy at the same directory where your document (tex file)
      is situated.
      Then instead of running LaTeX itself you have to run "libtex" as
      libtex filename
      ("filename" WITHOUT tex extension!!! For 2sorts.tex it is "2sorts")

      JOURNAL - name of the template
      PREFIX  - Biblio's home directory (where in the "formats" directory all
                formats are stored)

   B. biblist_html
      Makes an HTML formated list from your references database. I use (as you
      could see in refers.db) field URL for articles, which I have
      electronically. This script will run "biblist.pl" (see below) with --html
      key. This will produce nice html output with hyperlinks to your e-library
      (it will put prefix "--prefix" for all hyperlinks except starting already
      with "http:" )
      Of course, you can use "--prefix /home/usrname/base/" if you want to make
      all refs local.

   C. biblist
      The same as the previous one but output in TeX format. Uses biblist.pl
      without --html key.

   D. biblist.pl
      Makes a list of references from the whole database file. Can output in
      TeX file or HTML. Has switches: almost all, which biblio has plus --html.
      Does not have --uselist (it is always on), --start_refs, --end_refs.
      And, of course, it does not need an input TeX file! Therefore you
      always have to provide format file.
      If --html is not given, produces TeX file.
      HTML output needs special formats, of course. Example is in
      EXAMPLES/formats/formats_html.biblist

      Example of a format file for TeX output is in
      EXAMPLES/formats/formats.biblist

   E. bibcheck.pl  (added 17 Sep 2001)
      Checks your database (one ore more files containing references).
      Complains if it has:
       - records with empty LABEL or TYPE
       - several records with the same LABEL
      Has specific parameter "--noexit" which means "do not exit if
      found a record without a label, just go on". By default it will
      halt in this case.
      Parameter -e (or --errors) says "after how many errors exit the program"
      (default is 5)
      For instance:
```

```
       bibcheck.pl -e 10 -d myrefers.db onemorerefs.db
     or
       bibcheck.pl --errors 10 -d myrefers.db onemorerefs.db
     will process files "myrefers.db" and "onemorerefs.db" but exit if at
     some point number of errors will exceed 10.


That's all for the moment.

-------------------------------------

8. INSTALLATION GUIDE and SHORT INSTRUCTIONS

I decided to put this section at the very end because now, I hope, you know what
you are installing.

After unpacking the archive you have:

 * biblio.pl     - the main "executable"
 * dir formats   - format files (template) and
                   abbreviations file "abbr.tex". Ready for use.
 * biblist.pl; biblist.pl;  biblist_html; bibcheck.pl
                 - some additional scripts (see section 7).
 * dir EXAMPLES  - examples (see section 6).

and bash script "install" to help you to install.

1. Edit script "install" First two lines have to point to
   * PREFIX to the directory, where you want to install biblio
     (formats, examples, etc.). It must NOT exist yet.

   * BINDIR to the binaries directory, where it will put links to biblio and
     biblist

   Edit biblio.pl and ADDSCRIPTS/biblist.pl: in the first line it should be
   a full path to your Perl!!!

2. Run "./install". Then change dir to your PREFIX dir and check, that biblio.pl
   and everything in the ADDSCRIPT directory is executable. If not, do

   chmod a+x name_of_the_file


3. If installation script failed, you can do it yourself.
   Just create a directory for biblio.
   Copy there everything you found in the current directory (see above).
   Then

   ln -s biblio biblio.pl

   Chenge dir to your normal binaries dir and do

   ln -s biblio PREFIX/biblio
   ln -s biblist.pl PREFIX/ADDSCRIPT/biblist.pl

   where PREFIX has to be a real path (like /usr/local/biblio)

   Be sure about file permissions (everybody, or at least you, has to be
   able to execute!).

That's all. Now you have commands "biblio" and "biblist"
```



```
About usage of additional scripts. What I normally do is like this:

1. Copy libtex from ADDSCRIPTS to the directory where my current TeX document
is.

2. Edit it

   * JOURNAL is the name of the template
   * PREFIX  is the place where you keep biblio (and all its stuff)
   * REFS is your references database
   * FORMATS is a place where you keep format files (templates)
   * PROJECT is a name of your TeX file WITHOUT extension.
     Normally you do for file myarticle.tex
        libtex  myarticle
     and myarticle goes to PROJECT variable.
     But you can just put it like
     PROJECT=myarticle
     in the text of libtex and then type
        libtex
     only to compile your TeX document.

3. After running libtex (if everything goes right) you will have
   a file named "myarticle.dvi" and "myarticle.1.tex" and some more, which are
   not important.

Working version of libtex (it works even without any installation) is in the
EXAMPLES dir. Try it.
-----

About biblist - it produces a full references list from all the entries you have
in your database. Copy it somewhere and edit:
REFSTEX is the output file;
FORMATS is the format file. You can edit it, if you like.

Run biblist and you get references.tex file, and references.dvi file.
-----

About biblist_html - does the same what biblist does, but outputs in HTML.
About --prefix option you can read in 7.B and 7.C. Formats, of course, are
different from what you use for TeX output. I provide formats_html.biblist file
(find it in the "formats" directory) - you can easily edit it.

-------------------------------------

9. PERL TO C

If everything went OK and you are going to use biblio, I have an idea about
speeding it up. You can use "perlcc" to make a real binary from Perl's script.
Sometimes it is a good idea, sometimes not. What I found: binary of biblio
is 50% (yeah, yeah - 50!) faster than Perl-script. So it is worthy of trying!
Do:

   perlcc -o biblio.exe  biblio.pl

and you will get biblio.exe executable (it will take some time - translation to
C and then compilation).

If you succeeded, make a link called "biblio"  to biblio.exe (at the same
directory, where you have biblio.pl; so, you have to erase an old link biblio
pointing to biblio.pl).

All the rest is the same.
```

If you are interested in C sources, try

```
   perlcc -sav -o biblio.exe  biblio.pl
```

It will save sources for you! Just great! Althoug I don't think, this C-source is really readable! :)

----

10. TO DO

A lot of things are very raw. Only a couple of people tried it and I cannot say it was sufficiently tested. So if you discover any bug or have any suggestion write directly to me (rstepanyan@yahoo.com).

This script now supports only very primitive way of citation: each citation is numbered and this number is its identifier. So, citation in the form "Smith(1990a)" are impossible. That is, probably, the first thing to do (This kind of citations is used in Rev. of Modern Physics, for instance).

I do not have too much time right now, but I am constantly thinking about it.

----

11. Known Bugs

11.A. Lost Lines (for ver.1.2 only)

From ver.1.2 there is no points and commas at the beginning of the line. Like in the piece of text

```
    Was shown in work
    \cite{lab2}. Blah-Blah-Blah
```

if we use --movepoint option we get

```
    Was shown in work
    .\cite{lab2} Blah-Blah-Blah
```

(and ver.1.1 really did it this way).
But it should be (and from ver.1.2 it IS)

```
    Was shown in work.\cite{lab2} Blah-Blah-Blah
```

That is exactly what is fixed now. Although it adds a new bug: if you have

```
    Was shown in work
    % Blah-Blah-Blah
    \cite{lab2}. Blah-Blah-Blah#
```

then one line is LOST!!!! (Guess why!)

Oh, now it is really the EOF!