CrossTEX Tutorial

Emin Gün Sirer and Robert Burgess

This tutorial will show you everything you need to know about CrossTEX. It assumes basic familiarity with BibTEX. *You should have CrossTEX installed.*

CrossTEX is a modern bibliography typesetting tool that works in conjunction with LaTeX. It first builds an object hierarchy based on the bibliography database. Then it parses the text at hand to determine which objects are being cited. Then it formats these objects according to the style selected in the document, as modified by the command line options. It then produces a references section that LaTeX can incorporate into the original document.

1 Quick start

First, make sure CrossTEX is installed. Then, where you used to type:

```
$ latex paper  # Generate the .aux file
$ bibtex paper # Generate the .bbl file
$ latex paper  # Incorporate the bibliographic information
$ latex paper  # Get the labels right
```

Instead use:

```
$ latex paper    # Generate the .aux file
$ crosstex paper # Generate the .bbl file
$ latex paper    # Incorporate the bibliographic information
$ latex paper    # Get the labels right
```

CrossTEX is backwards-compatible with BibTEX and supports the standard `abbrv`, `alpha`, `full`, and `plain` bibliography styles.

2 Defining Objects: Inside the `.xtx` file

2.1 Objects

Everything in CrossTEX is an object. Every object has a *key* that can be used to refer to it, and *fields* containing values. Here are some objects:

```
@month{sep, name = "September"}

@location{rio, name = "Rio de Janeiro, Brazil"}
```

```
@author{egs, name = "Emin {G\"un} Sirer"}

@article{mypaper,
   author = egs,
   title = "This is My Paper",
   journal = "Journal of Improbable Results",
   address = rio,
   year = 2018,
   month = sep
}
```

The first line defines a `month` object, henceforth known as `sep`, that has a single field called `name`, which consists of the string "September". From here on, other objects can simply refer to `sep` wherever a month is called for, and they will be referring to this object. The second line defines a `location` object named `rio`, while the third line defines an `author` object whose name requires complicated LaTeX punctuation to format properly. The final entry defines an `article`, published in Rio de Janeiro in September. Note how it refers to the previous objects by their keys. The fields of `mypaper` end up as though it had been defined thus:

```
@article{mypaper,
   author = "Emin {G\"un} Sirer",
   title = "This is My Paper",
   journal = "Journal of Improbable Results",
   address = "Rio de Janeiro, Brazil",
   year = 2018,
   month = "September"
}
```

Objects can be given multiple keys, as well. Take for example the following author:

```
@author{rama, name="Venugopalan Ramasubramanian"}
```

However, Rama uses Venu and Arun as names for his alter egos. So we can define his object as follows:

```
@author{rama = arun = venu, name="Venugopalan Ramasubramanian"}
```

Thereafter, the following are equivalent.

```
author = "rama and egs"
author = "arun and egs"
author = "venu and egs"
```

2.2 Representation

Every kind of object, such as `month`, `location`, `author`, and `article`, knows how to convert itself into a string suitable for inclusion in the references section of a scholarly publication. For example, when `mypaper` referred to `sep` in the example, the actual value assigned was "September". Some objects, such as `mypaper` itself, will produce entire bibliography entries when referred to. In fact, when generating bibliographies, `crosstex` simply prints out the string representations of all the objects cited in the document. Each object takes note of options passed into CrossTEX and generates a string representation accordingly.

Simple, named objects, such as `month`, `location`, `author`, and others, have two forms: A long form and a short form. The object `sep` could be defined:

```
@month{sep,
  name = "September",
  shortname = "Sept."
}
```

By default, long names are used when generating bibliographic entries. If, however, the option `--short month` were given, the same month object would be shown as "Sept." instead of "September". If only a name (no shortname) is specified, or, conversely, if only a shortname but no name is specified, the name given will be used in all cases. All named objects follow this pattern.

However, because person names are so complicated, `author` objects are somewhat magical. If `--short author` is specified, authors will be represented with initials and a last name—the object `egs` would be represented "E. G. Sirer". If a `shortname` field is explicitly given for an author, that takes precedence and can be used for special cases. By default, CrossTEX is aware of many kinds of names and can correctly handle suffixes and last name modifiers such as Jr., Sr., III, IV, von, van, de, bin, and ibn. Just write the names out in full in their natural order, and CrossTEX will render them properly, including such entries as:

```
@author{rvr, name = "Robbert van Renesse"}
@author{ldv, name = "Louis de Vargas, III"}
```

The `author` field is also somewhat special. Users of BIBTEX are familiar with specifying multiple authors as follows:

```
@inproceedings{credence,
  title = "{Experience with an Object
            Reputation System for
            Peer-to-Peer Filesharing}",
  author = "Kevin Walsh and Emin {G\"un} Sirer",
  ...
}
```

Having looked up the rather non-trivial escape sequence to get the umlaut correct, and having made sure that all names are spelled correctly, there is no need to repeat or cut-and-paste the same information over and over again. The following sequence has the same effect as the previous one, and is much easier to maintain:

```
@author{kwalsh, name = "Kevin Walsh"}
@author{egs, name = "Emin {G\"un} Sirer"}
@inproceedings{credence,
  title = "{Experience with an Object
            Reputation System for
            Peer-to-Peer Filesharing}",
  author = "kwalsh and egs",
  ...
}
```

In all other contexts, string literals and object references are quite different: `month = sep` refers to the `sep` object, which carries additional fields and is rendered differently depending on the context and options, while `month = "sep"` always generates the literal string "sep".

2.3 Under the hood: References

Referring to an object, as in `month = sep`, assigns the stringified value of the object `sep`, in this case "September", to the `month` field of the object. In addition, it triggers something else to happen after all the fields have been assigned: Any required or optional fields in the referring object that do not yet have values will try to inherit them from the referenced object `sep`, if it has them. The actual `month` objects defined in the included `dates.xtx`, for example, define `monthno` fields in addition to their names. This value will be pulled along into objects that refer to months, enabling such objects to be sorted by month number easily. Power-users of BibTeX will note that this mechanism is essentially equivalent to the `crossref` field in BibTeX. The main difference here is that CrossTeX supports this mechanism in a uniform manner across all object fields.

This can be combined with a new, advanced feature of CrossTeX, conditional fields, to greatly simplify object specifications. Here is an example:

```
@conference{nsdi,
  shortname = "NSDI",
  longname = "Symposium on Networked System Design and Implementation",
  [year=2006] address=SanJose, month=may,
  [year=2005] address=Boston, month=may,
  [year=2004] address=SF, month=mar,
}
```

This is a simple `conference` object whose short name is "NSDI" and long name is "Symposium on Networked System Design and Implementation". In

addition, it carries some conditional fields; that is, fields that are included in the object only if they match the data in the referring context. For instance, if the `year` field of the referring object is equal to 2006, the `nsdi` object additionally has the fields `address=SanJose, month=may`. In a different year, a different conditional might be triggered. Any, all, or none of the conditionals mentioned may be appropriate (although obviously in this case, the year can only have one value).

By itself, conditional fields are not very useful; their value in simplifying object references becomes apparent when they are used in context. Look at the following reference:

```
@inproceedings{credence,
  title = "Experience with an Object Reputation System for Peer-to-Peer Filesharing",
  author = "kwalsh and egs",
  booktitle = nsdi,
  year = 2006
}
```

Conditional fields defined in the `nsdi` object will define the address and month fields of the conference based on the reference, and the resulting `credence` object will know that it occurred in San Jose in May through inheritance. This allows paper citations to avoid common errors by allowing all conference dates and locations to be defined in one place, and inherited correctly, without typos, by all papers that appeared at that conference.

If one wanted to override field inheritance for whatever reason, it would suffice to specify, say, a different month for the `credence` object. Only those fields that are missing in the referring context are inherited. Thus explicitly assigned information has precedence.

2.4 Including other databases

Obviously, re-inventing `sep` and `SanJose` in every database would be exhausting. Instead, similar objects can be collected together—for example, the standard CrossTEX distribution provides `dates.xtx`, which defines English month names, and `locations.xtx`, which defines all locations at which a major computer science conference was held in the recent years. Such modules can be includes with the `@include` primitive. For example, here is a complete `.xtx` file based on the standard CrossTEX distribution:

```
@include conferences-cs

@author{egs, name = "Emin {G\"un} Sirer"}
@author{kwalsh, name = "Kevin Walsh"}

@inproceedings{credence,
  title = "Experience with an Object Reputation System for Peer-to-Peer Filesharing",
  author = "kwalsh and egs",
  booktitle = nsdi,
```

```
  year = 2006
}
```

This will search for `conferences-cs.xtx` or `conferences-cs.bib` and include it in the appropriate place before parsing the rest of the file. CrossTEX by default looks in a standard system directory and the directory containing the database or document being processed; additional search paths can be specified with the `--dir` option. The standard `conferences-cs.xtx` begins:

```
@include dates
@include locations

...
```

Thus, the `credence` object has access to well-defined locations, dates, and conference names.

On startup, `crosstex` will read in the database `standard`, which in the distribution pulls in the `dates` database and some information to help accurate formatting of titles. Having the dates available by default is necessary for backwards compatibility with BibTEX; the administrator may also edit the standard database in order to automatically include additional important files such as an institution-local bibliography.

It is possible for the same object to be defined multiple times under the same key (sometimes, this is inevitable when there are multiple bibliographic databases involved maintained by different entities). By default, CrossTEX will silently ignore such definitions as long as all versions of the object are identical. When two separate objects defined under the same key are not identical, it points to an inconsistency in the bibliographic database, which will cause CrossTEX to issue a warning. Passing CrossTEX the `--strict` flag will force it to issue such warnings even when the objects are identical, to help facilitate people who might want to maintain databases free of duplicate entries.

2.5 Extending objects

Occasionally it is useful to add information to an object that already exists. For example, say you have a paper to cite that appeared in USENIX 2006, but the system database only has the following information about the USENIX conference:

```
@conference{usenix = usenixg,
  shortname = "USENIX",
  longname = "USENIX Annual Technical Conference",
  [year=2005] address=Anaheim, month=apr,
  [year=2004] address=Boston, month=jun,
  [year=2003] address=SanAntonio, month=jun,
  [year=2002] address=Monterey, month=jun,
  [year=2001] address=Boston, month=jun,
  [year=2000] address=SanDiego, month=jun,
```

```
  [year=1999] address=Monterey, month=jun,
  [year=1996] address=SanDiego, month=jan,
}
```

Obviously, the best solution is to add the following line to the entry in the system conferences database:

```
  [year=2006] address=Boston, month=may,
```

However, you may not have permission to edit the database. Now there are two options: Cut-and-paste the usenix object into some local database with a new name so there is no conflict, or put the address and month directly into the paper's entry. Neither one is a good solution. What you want is to be able to extend the `usenix` object even though it is in another database you can't edit.

Enter the `@extend` primitive. The following solves the example:

```
@extend{usenix,
  [year=2006] address=Boston, month=may,
}
```

An `@extend` entry looks just like an object definition. However, rather than defining a new object, the object with the specified key is re-built with the information provided, inheriting its old fields with lower priority so that extended fields take precedence.

It is possible to create new aliases along the way. Simply list aliases in the exact same syntax as for object definition; all the aliases listed must either be a new, unused alias or refer to the same unique object. After the object is extended, all the aliases mentioned will be handles to refer to the newly-extended object. This can be useful for defining shorter, easier-to-remember names for database objects.

2.6 Default fields

When databases get exceptionally long and many elements have very similar fields—e.g., they are all in the same conference or have the same informative `category` field—you can make use of another special CrossTEX command, `@default`. For example, here is the beginning of the `usenix.xtx` database:

```
@include conferences-cs

@inproceedings{DBLP:conf/usenix/RuanP04,
  author    = {Yaoping Ruan and
               Vivek S. Pai},
  title     = {Making the "Box" Transparent: System Call Performance as
               a First-Class Result},
  booktitle = usenixg,
  year      = 2004,
  pages     = {1-14},
```

```
  ee        = {http://www.usenix.org/publications/library/proceedings/usenix04/t
ech/general/ruan.html},
  bibsource = {DBLP, http://dblp.uni-trier.de}
}

@inproceedings{DBLP:conf/usenix/CantrillSL04,
  author    = {Bryan Cantrill and
               Michael W. Shapiro and
               Adam H. Leventhal},
  title     = {Dynamic Instrumentation of Production Systems},
  booktitle = usenixg,
  year      = 2004,
  pages     = {15-28},
  ee        = {http://www.usenix.org/publications/library/proceedings/usenix04/t
ech/general/cantrill.html},
  bibsource = {DBLP, http://dblp.uni-trier.de}
}

...
```

With `@default`, it could be shortened:

```
@include conferences-cs

@default booktitle = usenixg
@default year = 2004
@default bibsource = {DBLP, http://dblp.uni-trier.de}

@inproceedings{DBLP:conf/usenix/RuanP04,
  author    = {Yaoping Ruan and
               Vivek S. Pai},
  title     = {Making the "Box" Transparent: System Call Performance as
               a First-Class Result},
  pages     = {1-14},
  ee        = {http://www.usenix.org/publications/library/proceedings/usenix04/t
ech/general/ruan.html},
}

@inproceedings{DBLP:conf/usenix/CantrillSL04,
  author    = {Bryan Cantrill and
               Michael W. Shapiro and
               Adam H. Leventhal},
  title     = {Dynamic Instrumentation of Production Systems},
  pages     = {15-28},
  ee        = {http://www.usenix.org/publications/library/proceedings/usenix04/t
ech/general/cantrill.html},
}

...
```

Later in the file are entries with different years. A new `@default` command takes precedence over the first:

```
...

@default year = 2003

@inproceedings{DBLP:conf/usenix/PadioleauR03,
  author    = {Yoann Padioleau and
               Olivier Ridoux},
  title     = {A Logic File System},
  pages     = {99-112},
  ee        = {http://www.usenix.org/events/usenix03/tech/padioleau.html},
}

@inproceedings{DBLP:conf/usenix/DouglisI03,
  author    = {Fred Douglis and
               Arun Iyengar},
  title     = {Application-specific Delta-encoding via Resemblance Detection},
  pages     = {113-126},
  ee        = {http://www.usenix.org/events/usenix03/tech/douglis.html},
}

...
```

As with field values inherited from references objects, field values inherited from `default` definitions have lower precedence. Any object that explicitly assigns a value to a field will override any `default` definitions in effect at that point in the bibliography.

2.7 Comments

Comments in CrossTEX can be accomplished in a number of ways. Simple comments that last until the end of the line are introduced with a `%` character. For example:

```
@include conferences-cs   % Because we need nsdi later on
```

More involved, potentially multi-line comments appear as their own kind of primitive:

```
@comment "This is a database for...
Yadda yadda...
Now I've said enough."
```

This syntax can also take advantage of the `{...}` form of strings in order to comment out whole objects or sets of objects, since braces are counted and matched correctly so that embedded strings don't accidentally end the comment.

```
@comment {
  @inproceedings{bad,
    title = "Some paper we want to temporarily comment out",
    author = "Somebody and Somebody Else",
    ...
  }
}
```

3 Citing References

Once you have defined your objects in the `.xtx` file, you may refer to them in your `.tex` file. Such references are known as citations, and are accomplished with the `\cite` command in LaTeX. CrossTeX supports two kinds of citations, both backwards compatible with standard LaTeX citations.

3.1 Plain Citations

The first type are plain citations based on an object key. Plain citations simply take a comma-separated list of object keys, and cite the objects whose keys, specified in the XTX file as the first item following the object definition, match the cited key. For instance, given the definitions above, the following are examples of plain citations:

```
Credence~\cite{credence} provides a reputation
system for peer-to-peer systems.  These two
papers~\cite{DBLP:conf/usenix/PadioleauR03,DBLP:conf/usenix/DouglisI03}
appeared at the Usenix annual conference.
```

The key used in a plain citation must match, exactly, the key used in the object definition. The matching is case sensitive, so "foo" and "FOO" refer to different objects.

Recall that CrossTeX enables an object to appear under multiple keys. This aliasing can be done for any object and can be used anywhere in the database. The `.tex` file can cite any object by any one of its synonymous keys. There is a strange quirk with the use of synonymous keys stemming from a design error in LaTeX, which users should keep in mind: LaTeX assumes that each object has only one key, and thus citing the same object under two different keys would require it appearing twice in the references. Therefore, authors must be careful to cite each paper by only one of its aliases. Fortunately, it does not matter which alias is used in the document, so long as it is consistent, and it is easy for CrossTeX to detect when multiple aliases are being used, so an error message will appear.

Overall, there really are not that many frills to plain citations. They work exactly the way one would imagine they would. Their big drawback, however, is that you need to remember the precise key for every object you want to cite. Often, this requires browsing database files, searching for author names and

keywords in the title so you can figure out whether you named the key "credence" or "credence_nsdi04" or "nsdi04_credence". Even though the standard libraries that come with CrossTEX follow the uniform naming rule from the DBLP database, figuring out the uniform name still requires knowing the authors and the year, which often requires a Google search. To make the citation process even easier and simpler, CrossTEX supports a second kind of citation, where the user need not recall the object key precisely.

3.2 Constrained Citations

The second kind of citation that CrossTEX supports is known as a *constrained citation*. Constrained citations enable the user to cite a paper by specifying pieces of information about the reference that uniquely identify it. For instance, suppose you want to reference that paper I wrote in 1999 on how to split up virtual machines, and you remember that it appeared at SOSP. You could search your database for some partial terms that appear in the entry (e.g. 1999, sosp, sirer), copy the key for the entry, and issue a plain citation using that precise key. This is what many BIBTEX users do without thinking. But it is a lot of pointless boring work, and computers were supposed to automate boring tasks. That's where constrained citations come in.

A constrained citation begins with an exclamation point, and specifies a series of colon-separated terms that identify the reference being cited. Some examples of constrained citations are:

```
\cite{!author=sirer:title=virtual:year=1999}
\cite{!author=sirer:title=virtual:title=machines:year=1999}
\cite{!author=sirer:author=walsh:year=2006}
```

Colons separate constraints. Each constraint identifies a field that the reference must have, as well as a string that should appear somewhere within that named field.

Each string in a constrained citation is checked for a partial match in the corresponding field. So "author=smith" will match both "Smith" and "Smithson."

Sometimes, there are multiple constraints that apply to the same field. Specifying the same field multiple times, as in the second and third examples above, is perfectly acceptable, but gets tedious. So CrossTEX provides a way to specify multiple constraints for the same field; every word separated by a "-" sign is treated as a separate constraint. So the examples above can be shortened down to:

```
\cite{!author=sirer:title=virtual:year=1999}
\cite{!author=sirer:title=virtual-machines:year=1999}
\cite{!author=sirer-walsh:year=2006}
```

Multiple constraints within a given field are not ordered and can appear anywhere in the string, so "virtual-machines" will match "virtual machines," as well as "machines virtual," and even "building a machineshop virtually."

Several shorthands make constrained citations even easier to specify by providing defaults for fieldnames. If the fieldnames are missing, the first constraint defaults to "author." The second constraint defaults to "title" if the value is not numeric; if it is, it defaults to "year." Finally, the last constraint defaults to "year." So the examples above can be shortened even further:

```
\cite{!sirer:virtual:1999}
\cite{!sirer:virtual-machines:1999}
\cite{!sirer-walsh:2006}
```

Two caveats are worth remembering about constrained citations. First, the citation needs to be uniquely identifiable. If the constraints you specified match more than one object, CrossTEX will print an error and identify the matching objects. You can then specify more constraints until you have nailed down the reference you had in mind or switch to a plain citation. Second, due to a limitation in LATEX mentioned above for plain citations, referring to the same paper through different constraints (e.g. "!sirer:virtual:1999" and "!sirer:virtual-machines:1999") will cause an error so the paper does not appear twice in the references section. For each paper, you should figure out the constraints you had in mind and stick to them throughout your document.

Overall, constrained citations are a very convenient way to cite papers without having to look anything up. They fit naturally to the way people recall citations. The concept was entirely lifted from Norman Ramsey's `nbibtex` system.

3.3 Citation Appearance

How the citation itself appears is controlled by the citation style, and is controlled by options specified to crosstex either in the LATEX file or passed on the command line during invocation. The argument to the `--cite-by` option determines how the citations appear in the body of the text. There are three possible arguments to `--cite-by`.

`numeric` produces citations that appear like this "[1]". The numbers correspond to the location of the entry in the references section. Another option determines how the references section is sorted (e.g. in the order cited, alphabetized by author, or sorted according to any field of choice), and thus affects the particular number used to refer to a particular reference.

`initials` produces citations that appear like this "[WS04]". The particular rule used to derive the initials from author names is somewhat complex, but roughly speaking, the citation string consists of the first initials of the authors last names, appended with the year of publication. If there is a single author, then the first three letters of the author's last name is used instead. A paper by Sirer in 2006 would be cited as "[Sir06]" under this scheme. If there are five or more authors, the first three initials are appended with a "+" sign and the year. For instance, a paper by Aardvark, Dewey, Chethem, and Howe would be cited as "[ADCH06]", but if Aardvark and friends sign on Elvis as a coauthor, the citation string becomes "[ADC+06]". Finally, last name modifiers (such as

"van") are preserved in lower case. A paper by Sirer and van Renesse would be cited as "[SvR07]".

`fullname` produces citations that appear like this "[Walsh and Sirer 06]". The last names appear in full for references authored by up to two authors. A paper by Dewey, Chethem and Howe would be cited as "[Dewey et al. 06]". Fullname citations are the most readable and should be used whenever possible.

4 The References Section

CrossTEX provides many options that enable the user to control the appearance of the references within a document. This section describes various options that can be passed to the `crosstex` tool for achieving the precise formatting desired.

4.1 Invoking CrossTEX

In its simplest invocation, `crosstex` takes a file name, e.g. `crosstex` *file1*. Any included files are found in a search path containing the directory with the file being processed and a central system directory (e.g. `/usr/local/crosstex/lib`); this search path can be extended with the `--dir` option. Extensions (`.aux`, `.xtx`, `.bib`) will be added if necessary to find the file. Each output will always appear in the same directory as the file processed, under the same name but with the extension changed to `.bbl`. `crosstex` will exit with an error code if any warning or error messages were printed.

If `crosstex` is invoked as `xtx2bib` or the `--xtx2bib` option is given, the output extension will be `.bib`, and the bibliographic information will be back-converted to plain BIBTEX.

If `crosstex` is invoked as `bib2xtx` or the `--bib2xtx` option is given, the output extension will be `.xtx`, and the bibliographic information will be output using CrossTEX's advanced features where possible. Currently, it is possible to use the `--heading` and `--reverse-heading` as usual to specify any hierarchy of fields to pull out with `@default` statements. This feature can be very convenient for converting old BIBTEX databases to CrossTEX, but might lose some information if used on an already optimized CrossTEX database.

If `crosstex` is invoked as `xtx2html` or the `--xtx2html` option is given, the output extension will be `.html`; some style information will be changed as appropriate for formatting a web bibliography, and the output will be wrapped into a LATEX document and translated into HTML by piping it through `hevea`. Sometimes it is necessary to run this more than once to get labels right, as with LATEX; `hevea` will print an appropriate message if this is necessary. By default, the style used for HTML is pretty non-traditional, but can be overridden by further options: `xtx2html --style plain` *file* looks nice and tame.

A number of options can be specified to change the style of the bibliography to override or tweak that specified by a document.

4.2 Optional fields

Optional fields in any object can be "turned off" with the `--no-field` option. The option can be specified multiple times with different fields, and if any of those fields are specified in the database where they are optional, the fields will be ignored and left blank. As a result, for example, to avoid displaying page numbers is as simple as `--no-field pages`.

4.3 Abbreviation

The `--short` option allows many kinds of objects to be abbreviated in the bibliography. For example, to use shortened month names ('Jan.', 'Feb.') instead of long ones ('January', 'February'), simply use the option `--short month`. This allows the creators of the database to specify both month names just once, refer to the relevant `month` objects in their entries, and the formatting of month names to be consistently chosen when the bibliography is formatted.

Anything with a name can be abbreviated this way—so a conference can be shortened from "Networked Systems Design and Implementation" to "NSDI" when under the space crunch or filled back out later with a simple option. Databases mention each name only once, and, even more importantly, what name to use is left to the document and the user and is not imposed on the database maintainer.

Objects that can be shortened include `author`, `conference`, `conferencetrack`, `country`, `journal`, `month`, `state`, `string`, and `workshop`.

4.4 Authors

Author names can be complicated, and are the source of much confusion in BibTeX. The same author might appear with a middle name, without a middle name, last name first, with abbreviated first names, mis-spelled, with different combinations of accents, and so forth.

In CrossTeX, the database maintainer can enter the name just once in an author object and control the way it is formatted via options. The `--short author` option generates abbreviated author names automatically if an author doesn't have an explicitly mentioned short name, and CrossTeX is careful to handle complicated names with accents and modifiers correctly when abbreviating or generating citation keys.

The option `--last-first` causes the first author in each list to be formatted 'Last, First' instead of 'First Last'. CrossTeX does the Right Thing with modifiers here, too. When author names are capitalized with `--capitalize author`, CrossTeX carefully works around LaTeX commands and accents to produce clean-looking names.

4.5 Capitalization

Any object that can be abbreviated with `--short` can be coerced to all upper case with `--capitalize`. For example, to cause authors to appear capitalized,

issue `--capitalize author`.

4.6 Titles

Title case is one of the most common inconsistencies when using BibTeX. Often, some papers are cited with lower-case titles, some are all upper-case, and some follow mixed title-case. Key acronyms (e.g. BGP) and proper nouns (e.g. Internet) are haphazardly capitalized, or not, depending on how diligent the author was when putting together the bibliographic database.

CrossTeX ensures that all titles follow the same uniform capitalization standard, even if they appear in a wild variety of styles in the database. The first letter of each word will become capitalized, the rest lower, the standard known as "titlecase". CrossTeX is very careful to ensure the titles come out looking "good"—words in StudlyCaps or ALLCAPS are retained as-is, LaTeX commands and anything in math mode are protected, compound words such as "Peer-to-Peer" are split into words, capitalized correctly, and re-assembled, and additionally a list of known phrases are carefully found and formatted. For example, any appearance of a string that is (ignoring case) equivalent to "USENIX" appears as "USENIX". These phrases are found at run-time by CrossTeX in `@titlephrase` commands, such as:

```
@titlephrase "USENIX"
@titlephrase "Linux"
```

The standard include files define certain common Computer Science phrases such as these, but they can appear anywhere in the `.xtx` file. Small words, such as "a", "an", "the", etc. are also handled specially: They are made lower-case except at the beginning of the title or after certain punctuation, such as long dashes or colons. These, too, are defined at run-time by `@titlesmall` commands:

```
@titlesmall "a"
@titlesmall "the"
```

Again, the standard include files define important English small words to start with.

An example title with the default might be "Aardvark: A System for Peer-to-Peer BGP Routing on the Internet".

With `--titlecase lower`, Only the first letter of the title and those following punctuation are capitalized, the rest put into lower-case. All of the special cases for the default title-case still apply. Thus, the example title would appear "Aardvark: A system for peer-to-peer BGP routing on the Internet".

With `--titlecase upper`, everything, even known phrases and small words, are put into upper-case thus: "AARDVARK: A SYSTEM FOR PEER-TO-PEER BGP ROUTING ON THE INTERNET". Commands and math-mode are still protected.

Finally, to allow titles to appear as they are specified in the database, use `--titlecase as-is`.

4.7 Proceedings

There are a variety of styles in use when citing papers at conferences. Some people prefer to precede the conference name with "In Proceedings of the ". These same people usually use "In Proc. of " when pressed for space. With `--add-proceedings`, CrossTEX will generate book titles for conferences beginning with "In Proceedings of", while `--add-proc` uses the shorter "In Proc." and without any options, only "In " is used for papers in conferences with proceedings.

For journal articles, the usual convention is to simply put the journal name in italics following the author names, and this is the default CrossTEX and BIBTEX behavior. Some people prefer to prepend "In " to the name of the journal; this can be accomplished with the `--add-in` option.

BIBTEX users affect these personal preferences by modifying the bibliographic database. Such changes are potentially disruptive and can introduce errors. CrossTEX enables such stylistic changes, which do not affect the underlying data, to be affected without modifying the database, and ensures that the choice will be applied consistently throughout.

4.8 Sorting and Headings

Sorting affects the order in which references will appear in the bibliography. By default, entries will be sorted by their citation keys, or by their authors and publication dates, depending on the citation style. The `--sort` option provides finer control over the sort order. By specifying `--sort` *field*, the database will be stably sorted by field; later specifying `--sort` *field2* will cause the bibliography to be sorted by field2, but the entries will still be sub-sorted by the first field. To sort in descending order, use `--reverse-sort` in the same way.

When processing large bibliographies, it can be nice to partition the entries into labeled categories. Specifying the `--heading` *field* option specifies a field to be used to divide the entries into sections. For example, `--heading year` will cause the entries to be grouped by year and given headings for each different year. (`--reverse-heading` will reverse the order in which the sections appear.) When converting a bibliography of personal publications to HTML, for example, it might be convenient to group by an information field such as `--heading category` to nicely organize the produced bibliography.

4.9 Hyperlinks

CrossTEX supports searching fields to find hyperlinks and presenting them in the references section. This is useful for any target format with hyperlinks, including PDF and HTML. Normally, no fields are treated as possible links, except when converting to HTML, when the list defaults to Abstract, URL, PS, PDF, HTML, DVI, TEX, BIB, FTP, HTTP, and RTF. A new field can be added with `-l` *field* or `--link` *field*; `--no-link` clears the list in order to disable link-finding or start over.

If any of the fields, case insensitive (e.g. `--link PDF` and `--link pdf` are equivalent), consists of a URL, it will appear at the end of the reference as a hyperlink with its label as the name of the field given to `--link` (e.g. the former would match the same field, but produce links labeled "PDF" and "pdf" respectively).

4.10 Abstracts and Keywords

Some detailed database entries might include a list of keywords related to the paper or even a complete abstract. By default, these fields are accepted but do not appear in the reference. With `--abstract`, abstracts will appear in blocks following the appropriate entries. The `--keywords` option invokes similar behavior for keyword lists. To explicitly set the defaults, `--no-abstract` and `--no-keywords` disable these extra blocks.

When converting to HTML, it may occasionally be desirable to make use of the dynamic nature of web sites and cause abstracts and keywords to appear as tooltip-like popups attached to each entry rather than taking up space on their own. The `--popups` option will cause the appropriate style manipulations (but does nothing without `--xtx2html` and at least one of `--abstract` or `--keywords`).

4.11 Putting the title first

Ordinarily, each entry begins with the author or editor first, then the title. A simple kind of re-ordering can be accomplished by specifying `--title-head`, which causes the title to come first and bold. This option can be negated with `--no-title-head`, but it is default only with `--style html` or `--xtx2html`.

4.12 Splitting up lines

Ordinarily, each entry takes up one logical line, which might wrap. The `--break-lines` option instead puts each major field (author, title, publication information, any hyperlinks, etc.) on its own line, in the same order they would have appeared on a single line. This can be combined with putting the title first with `--title-head` to cause the title to come on the first line all by itself, which is the default with `--style html` or `--xtx2html`. To explicitly cause entries to appear on a single logical line, use `--no-break-lines`.

4.13 Label appearance

By default, each entry in the references section is labeled with its citation key as it would appear in the document, e.g. "[1]" or "[WS04]". It is also possible, with the option `--blank-labels`, to leave entries un-labeled; this does not influence how citations appear in the document body in any way, but leaves out labels in the bibliography.

By itself, this option would probably produce a bibliography in which it is impossible to track down citations. However, it can be useful when converting a database to HTML, for example, when there is no document anyway and the labels look messy. Thus, leaving labels blank is the default with `--style html` or `--xtx2html`. To explicitly include labels, use `--no-blank-labels`.

4.14 Styles

Whole styles can be conveniently changed with the `--style` option, which overrides the style specified by the document with the specified style. When processing databases directly, the style defaults to `plain`; this option can force the database to be formatted with any style desired. CrossTEX styles are implemented as small, simple Python files in CrossTEX's path, and any style that happens to be installed at the site can be used with this option or the `\bibliographystyle` command in the document.

4.15 Inside the `.tex` file

The `.tex` source file never needs to know you're using CrossTEX, because it is completely backwards compatible with normal LATEX auxiliary files that note citations and styles. The `\bibliographystyle{`*foo*`}` command will cause the equivalent of the command-line option `--style` *foo*.

However, the `\bibliographystyle` command can cause other magic as well. Arbitrary command line options may be specified after the style name, separated by single spaces. For example:

```
\bibliographystyle{plain --add-in --add-proceedings --short author}
```

This allows documents to have fine-grained control over styling. Run-time options will still take precedence over document defaults.

4.16 Adding citations

Normally, only entries that are cited in the LATEX document appear in the references section. When processing a database directly, the default is for all entries to appear. These behaviors can be manipulated at run-time with the `--cite` option. It behaves exactly as a LATEX `\cite` command, including the use of the asterisk (`--cite *`, modulo shell-escaping) to cite all entries, and constrained citation. When processing LATEX documents, this adds entries to the references; when processing databases directly, this overrides the default to cite everything and cites only the entries specifically mentioned on the command line.

4.17 Manipulating `crosstex`

Numerous options help control or debug `crosstex` itself. The `--version` option will cause `crosstex` to print out its version and exit. To get a list of all supported options and brief descriptions, use `--help` or `-h`.

Two options change the level of error reporting. By default, only errors that will definitely change the appearance of the bibliography are produced. With `--strict`, more warnings, such as for unknown fields or other problems, will be printed. With `--quiet`, on the other hand, no errors or warnings will appear at all.

The `--dump` option provides very detailed debugging. Processing will continue exactly as normal, but at the end, any kinds of objects specified to `--dump` will be listed as output. For example, to list all the `author` objects defined in a bibliography *foo*, `crosstex --dump author foo` would process foo and print a list.

Two additional, non-object dumps are permitted. With `--dump file`, `crosstex` will print a trace of the path to every database it processes; this allows one to examine whether it is choosing the right files based on the search path and explore what standard databases are being automatically pulled in. The various title phrases and small words that control title capitalization can be dumped with `--dump titlephrase`.

5 Extending CrossTEX

CrossTEX is designed to be easy to extend with very trivial knowledge of Python. Before continuing, it is very important to have a look at the standard object types and fields in Appendix A, which are already supported by CrossTEX. New objects or fields are defined by editing the standard objects module `crosstex.objects`, which is typically installed as `/usr/local/crosstex/lib/crosstex/objects.py` or similar.

To create a new field for a particular object type, find its definition (e.g., the section defining the `string` object begins `class string`). Most objects already define some fields; simply copy that syntax for your own field. To create an entirely new class *foo* which is identical to a current one named *bar*, add the following to the end of the list of objects:

```
class foo(bar):
    pass # 'pass' is only necessary if no fields are defined.
```

Fields are defined as optional or required by assigning them the values `OPTIONAL` and `REQUIRED`, respectively. To make an optional field required or a required field optional, simply assign it the new value in the class where you want the change. To allow a field to inherit its value from another field in the same object if left blank, assign a string containing the name of the other field. A list containing `OPTIONAL`, `REQUIRED`, and one or more string field names will be processed and define several sibling fields and the given requirement level. For example, given:

```
class foo(bar):
    baz = REQUIRED
    blah = OPTIONAL
    quux = [REQUIRED, 'baz', 'blah']
```

This defines a new kind of object named *foo*, which behaves the same as *bar*; additionally, the 'baz' field is required, the 'blah' field is optional, and the 'quux' field is required but if unspecified will try to take its value from 'baz' or 'blah' in that order.

Styles are defined in small Python modules in the `style` directory in the same place you found `objects.py`. There you will find the default styles, `plain.py`, `full.py`, etc. Styles are built up from small filter functions mainly defined in `crosstex.objects`. Each field is filtered through four phases: Production, in which an initial value is generated from the object itself; List filtering, if the value is a list (otherwise irrelevant); List formatting, to turn a list into a string for the final step; and Filtering, in which the string is run through zero or more filters to come up with a final value. Look at the examples to see the syntax for hooking filters to each of the four stages in given contexts and for given types. Take as a simple starter the following statements:

```
misc._addproducer(emptyproducer, 'label')
conference._addfilter(proceedingsfilter, 'value')
misc._addfilter(emphfilter, 'fullpublication', 'booktitle')
```

The first states that the label attribute of any object derived from `misc` can be produced by `emptyproducer` if it returns anything other than `None`. (`emptyproducer` is defined in `crosstex.objects`.) The second statement causes the value of `conference` objects (and objects derived from `conference`) to be filtered through a filter that, in this case, prepends 'Proceedings of the' to the value. The last statement filters the 'booktitle' field of `misc`-derived objects, but only when included as part of the 'fullpublication' field (which happens to be a virtual field defined solely by attaching producers to it). It is important to note that filters and producers are applied starting from the most recent, so later producers will take precedence and later filters will be nested inside earlier filters. The standard styles are well-commented and should provide a good start.

Happy hacking!

A Standard object types

These are the kinds of objects CrossTEX knows about by default. For information about extending this notion, see Extending CrossTEX in Section 5.

`string` REQUIRED: `name` and/or `shortname` (`longname` is an alias for `name`.)
RELEVANT ARGUMENTS: `--short`

`author` As `string`, except: OPTIONAL: `address`, `affiliation`, `email`, `institution`, `organization`, `phone`, `school`, `url`

`state` As `string`, except: OPTIONAL: `country`

`country` As `string`.

`location` OPTIONAL: `city`, `state`, `country`

`month` As `string`.

`journal` As `string`.

`newspaper` As `journal`.

`misc` OPTIONAL: `abstract`, `address`, `affiliation`, `annote`, `author`, `bib`, `bibsource`, `booktitle`, `category`, `chapter`, `contents`, `copyright`, `crossref`, `doi`, `dvi`, `edition`, `editor`, `ee`, `ftp`, `howpublished`, `html`, `http`, `institution`, `isbn`, `issn`, `journal`, `key`, `keywords`, `language`, `lccn`, `location`, `month`, `monthno`, `mrnumber`, `note`, `number`, `organization`, `pages`, `pdf`, `price`, `ps`, `publisher`, `rtf`, `school`, `series`, `size`, `title`, `type`, `url`, `volume`, `year` RELEVANT ARGUMENTS: `--cite-by`, `--titlecase`, `--link`, `--abstract`, `--keywords`

`article` As `misc`, except: REQUIRED: `author`, `title`, `journal`, `year` RELEVANT ARGUMENTS: `--add-in`

`newspaperarticle` As `article`, except: (`newspaper` is an alias for `journal`)

`book` As `misc`, except: REQUIRED: `author` and/or `editor`, `title`, `publisher`, `year`

`booklet` As `misc`, except: REQUIRED: `title`

`inbook` As `misc`, except: REQUIRED: `author` and/or `editor`, `title`, `chapter` and/or `pages`, `publisher`, `year`

`incollection` As `misc`, except: REQUIRED: `author`, `title`, `booktitle`, `publisher`, `year`

`inproceedings` As `misc`, except: REQUIRED: `author`, `title`, `booktitle`, `year` RELEVANT ARGUMENTS: `--add-proceedings`, `--add-proc`

`manual` As `misc`, except: REQUIRED: `title`

`thesis` As `misc`, except: REQUIRED: `author`, `title`, `school`, `year`

`mastersthesis` As `thesis`.

`phdthesis` As `thesis`.

`patent` As `misc`, except: REQUIRED: `author`, `title`, `number`, `month`, `year`

`proceedings` As `misc`, except: REQUIRED: `title`, `year`

`collection` As `proceedings`.

`techreport` As `misc`, except: REQUIRED: `author`, `title`, `institution`, `year`

`unpublished` As `misc`, except: REQUIRED: `author`, `title`, `note`

`conference` As `string`, except: OPTIONAL: `address`, `crossref`, `editor`, `institution`, `isbn`, `key`, `keywords`, `language`, `location`, `month`, `publisher`, `url`, `year`

`conferencetrack` As `conference`, except: OPTIONAL: `conference`

`workshop` As `conferencetrack`.

`rfc` As `misc`, except: REQUIRED: `author`, `title`, `number`, `month`, `year`

`url` As `misc`, except: REQUIRED: `url` OPTIONAL: `accessyear`, `accessmonth`