kicost Documentation

Release 0.1.43

XESS Corporation

May 03, 2018

Contents

Contents:

CHAPTER 1

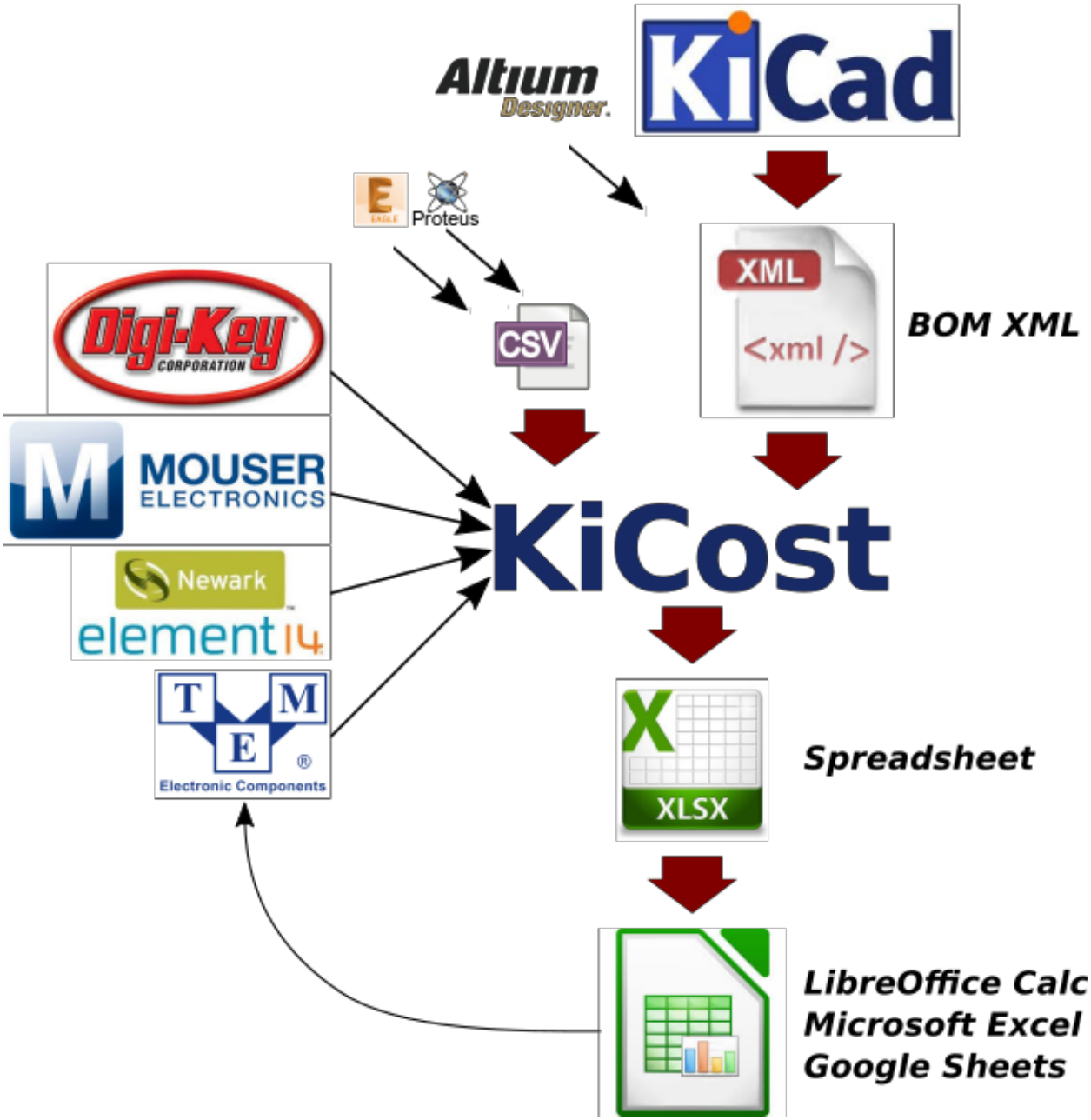
KiCost

KiCost is intended to be run as a script for generating part-cost spreadsheets for circuit boards developed with KiCad.

- Free software: MIT license
- Documentation: https://xesscorp.github.io/KiCost.

1.1 Features

- Processes the BOM XML file from your KiCad schematic to create a part-cost spreadsheet by scraping the web sites of several popular distributors for price and inventory data. (You can also enter your own quantity-adjusted pricing data for specialized parts or those not found at the supported distributors.);
- Processes also BOM files from Altium, Proteus, Eagle, Upverter and hand made CSVs;
- The spreadsheet contains quantity-adjusted pricing from each distributor for individual parts and the total board;
- Enter the number of boards to be built in a spreadsheet cell and all the pricing for the total board and individual parts is updated;
- The spreadsheet also shows the current inventory on-hand for each part at each distributor;
- Enter the quantity of each part that you want to purchase from each distributor and lists of part numbers and quantities will appear in formats that you can cut-and-paste directly into the website ordering page of each distributor.


Altium Designer
KiCad
Proteus
CSV
XML
<xml />
BOM XML
Digi-Key CORPORATION
MOUSER ELECTRONICS
Newark
element14
T M E
Electronic Components
KiCost
XLSX
Spreadsheet
LibreOffice Calc
Microsoft Excel
Google Sheets

CHAPTER 2

Installation

This is a Python package, so you'll need to have Python installed to use it. If you're using linux, you probably already have Python. If you're on Windows, you can download a Python installer from Anaconda , Active State , or even WinPython .

Once you have Python, you can install this package by opening a terminal window and typing the command:

```
$ easy_install kicost
```

Or:

```
$ pip install kicost
```

Note that if you install KiCost using `pip` on a Windows system running Python 2.7, using the default option that web scrapes with parallel processes may cause **MANY** errors. You can avoid this problem by:

- using `easy_install` to install KiCost, or
- use the `-s` KiCost option to serialize the web scraping.

On Linux, to install KiCost on python3, use:

```
$ pip3 install kicost
```

If you desire to test the new code version, putting you hands on new (beta) feature, you could install KiCost by:

    $ pip install git+https://github.com/xesscorp/KiCost.git

CHAPTER 3

Usage

KiCost's main use is generating part-cost spreadsheets for circuit boards developed with KiCad as follows:

1. For each part in your schematic, create a field called `manf#` and set the field value to the manufacturer's part number. (You can reduce the effort of adding this information to individual parts by placing the `manf#` field into the part info in the schematic library so it gets applied globally.) The allowable field names for part numbers are:

```
mpn          pn           p#
part_num     part-num     part#
manf_num     manf-num     manf#
man_num      man-num      man#
mfg_num      mfg-num      mfg#
mfr_num      mfr-num      mfr#
mnf_num      mnf-num      mnf#
```

2. Output a BOM from your KiCad schematic. This will be an XML file such as `schem.xml`.

3. Process the XML file with KiCost to create a part-cost spreadsheet named `schem.xlsx` like this:

```
kicost -i schem.xml
```

4. Open the `schem.xlsx` spreadsheet using Microsoft Excel, LibreOffice Calc, or Google Sheets. Then enter the number of boards that you need to build and see the prices for the total board and individual parts when purchased from several different distributors (KiCost currently supports Digi-Key, Mouser, Newark, Farnell, RS and TME). All of the pricing information reflects the quantity discounts currently in effect at each distributor. The spreadsheet also shows the current inventory of each part from each distributor so you can tell if there's a problem finding something and an alternate part may be needed.

5. Enter the quantity of each part that you want to purchase from each distributor. Lists of part numbers and quantities will appear that you can cut-and-paste directly into the website ordering page of each distributor.

3.1 Examples

Most people just want some examples of using KiCost so they don't have to read a bunch of documentation, so here they are!

To create a cost spreadsheet from an XML file exported from KiCad:

```
kicost -i schem.xml
```

To create a cost spreadsheet from within KiCad, use the `Tools` => `Generate Bill of Materials...` menu item and then enter the following in the *Command line* field:

```
kicost -i %I
```

To create a cost spreadsheet direct from the KiCad using the user definitions (by graphical interface last runned): To create a cost spreadsheet from within KiCad using the previous, use the `Tools` => `Generate Bill of Materials...` menu item and then enter the following in the `Command line` field:

```
kicost -i %I --user
```

To place the spreadsheet in a file with a different name than the XML file:

```
kicost -i schem.xml -o new_file.xlsx
```

To overwrite an existing spreadsheet:

```
kicost -i schem.xml -w
```

To get costs from only a few distributors:

```
kicost -i schem.xml --include digikey mouser
```

To exclude one or more distributors from the cost spreadsheet:

```
kicost -i schem.xml --exclude digikey farnell
```

To include parts that are only used in a particular variant of a design:

```
kicost -i schem.xml --variant V1
```

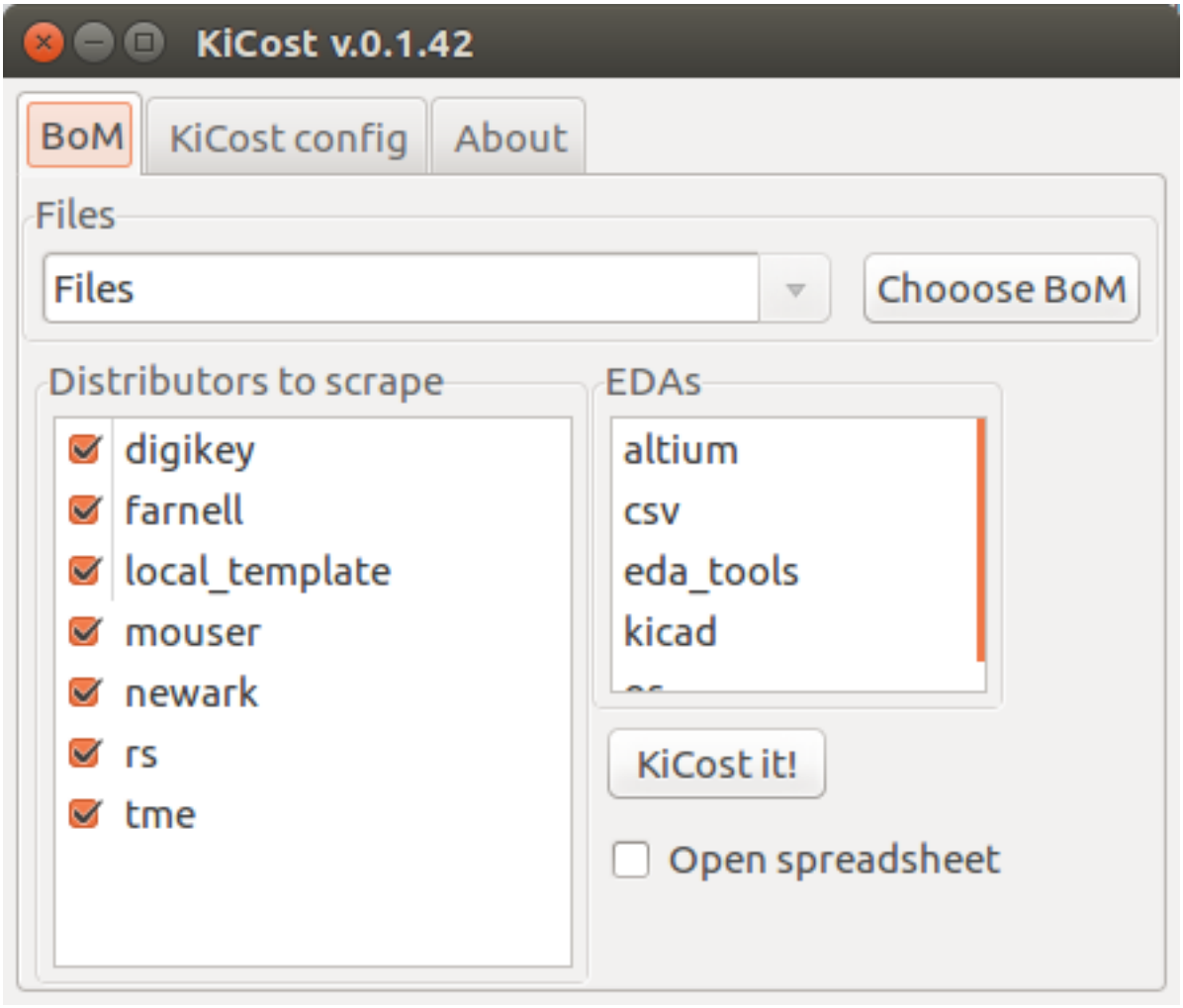
To create a cost spreadsheet from a CSV file of part data:

```
kicost -i schem.csv --eda_tool csv
```

To read and merge different projects BOMs, even those from different EDA tools:

```
kicost -i bom1.xml bom2.xml bom3.csv -eda kicad altium csv
```

To access KiCost through a graphical user interface, just use the *kicost* command without parameters.



3.2 Custom BOM list

In addition to XML files output by EDA tools, KiCost also accepts CSV files as a method for getting costs of preliminary designs or older projects. The format of the CSV file is as follows:

1. A single column is interpreted as containing manufacturer part numbers.
2. Two columns are interpreted as the manufacturer's part number followed by the part reference (e.g., `R4`).
3. Three columns are interpreted as the quantity followed by the part number and reference.

You can also arrange the columns arbitrarily by placing a header in the first line of the CSV file that labels the particular columns as manufacturer's part numbers (`manf#`), quantities (`qty`), and part references (`refs`).

3.3 Custom Part Data

The price breaks on some parts can't be obtained automatically because:

- they're not offered by one of the distributors whose web pages KiCost can scrape, or
- they're custom parts.

For these parts, you can manually enter price information as follows:

1. Create a new field for the part named `kicost:pricing` in either the schematic or library.
2. For the field value, enter a semicolon-separated list of quantities and prices which are separated by colons like so:

   ```
   1:$1.50; 10:$1.00; 25:$0.90; 100:$0.75
   ```

   (You can put spaces and currency symbols in the field value. KiCost will strip everything except digits, decimal points, semicolons, and colons.)

You can also enter a link to documentation for the part using a field named `kicost:link`. The value of this field will be a web address like:

```
www.reallyweirdparts.com/products/weird_product.html
```

After KiCost is run, the price information and clickable link to documentation for the part are shown in a section of the spreadsheet labeled **Local**. If you want to associate the pricing and/or documentation link to a particular source or distributor, just place an extra label within the field key to indicate the source like so:

```
kicost:My_Weird_Parts:pricing
kicost:My_Weird_Parts:link
```

Then the pricing and documentation link for that part will appear in a section of the spreadsheet labeled **My_Weird_Parts**.

You can have as many sources for parts as you want, and a part may have multiple sources.

3.4 Part Grouping

KiCost groups similar parts together and places their information on a single line of the generated spreadsheet. For parts to be grouped, they must:

- come from the same library (e.g., "device"),
- be the same part (e.g., "R"),
- have the same value (e.g., "10K" but note that this **would not match** "10000" or "10K0"), and
- have the same footprint (e.g., "Resistors_SMD:R_0805_HandSoldering").

To reduce your effort, KiCost will also propagate pricing data among grouped parts. For example, if you place a hundred 0.1 uF decoupling capacitors in 0805 packages in a schematic, you need only assign a manufacturer's number and/or pricing data to one of them and it will be applied to the rest.

There are several cases that are considered when propagating part data:

- If only one of the parts has data, that data is propagated to all the other parts in the group.
- If two or more parts have data but it is identical, then that data is propagated to any of the parts in the group without data.
- If two or more parts in the group have `different` data, then any parts without data are left that way because it is impossible to figure out which data should be propagated to them.

It is possible that there are identical parts in your schematic that have differing data and, hence, wouldn't be grouped together. For example, you might store information about a part in a "notes" field, but that shouldn't exclude the part from a group that has none or different notes. There are three ways to prevent this:

1. Use the `--ignore_fields` command-line option to make KiCost ignore part fields with certain names:

```
kicost -i schematic.xml --ignore_fields notes
```

2. Use the `--group_fields` option to allow grouping of parts even if they have different field values, but then display the parts separately in the spreadsheet using a multiline cell. The following example will group parts that are identical except for having different footprints, but will display them individually:

```
kicost -i schematic.xml --group_fields footprint
```

3. Precede the field name with a ":" such as `:note`. This makes KiCost ignore the field because it is in a different namespace.

3.5 Parts With Subparts

Some parts consist of two or more subparts. For example, a two-pin jumper might have an associated shunt. This is represented by placing the part number for each subpart into the `manf#` field, separated by a ";" like so: `JMP1A45; SH3QQ5`. The `manf` (manufacture name) also allow this division, empty or replicate the last one (use "~" character to replicate the last one). Each subpart will be placed on a separate row of the spreadsheet with its associated part number and a part reference formed from the original part reference with an added "#" and a number. For example, if the two-pin jumper had a part reference of `JP6`, then there would be two rows in the spreadsheet containing data like this:

```
JP6#1  ...  JMP1A45
JP6#2  ...  SH3QQ5
```

You can also specify multipliers for each subpart by either prepending or appending the subpart part number with a multiplier separated by a ":". To illustrate, a 2x2 jumper paired with two shunts would have a part number of `JMP2B26; SH3QQ5:2`. The multiplier can be either an integer, float or fraction and it can precede or follow the part code (e.g. `SH3QQ5:2` or `2:SH3QQ5`).

3.6 Schematic Variants

There are cases where a schematic needs to be priced differently depending upon the context. For example, the price of the end-user circuit board might be needed, but then the price for the board plus additional parts for test also has to be calculated.

KiCost supports this using a `variant` field for parts in the schematic in conjunction with the `--variant` command-line option. Suppose a circuit has a connector, J1, that's only inserted for certain units. If a field called `variant` is added to J1 and given the value V1, then KiCost will ignore it during a normal cost calculation. But J1 will be included in the cost calculation spreadsheet if you run KiCost like so:

```
kicost -i schematic.xml --variant V1
```

In more complicated situations, you may have several circuit variants, some of which are used in combination. The `--variant` option will accept a regular expression as its argument so, for example, you could get the cost of a board that includes circuitry for both variants V1 and V2 with:

```
kicost -i schematic.xml --variant "(V1|V2)"
```

A part can be a member of more than one variant by loading its `variant` field with a list such as "V1, V2". (The allowed delimiters for the list are comma (,), semicolon (;), slash (/), and space ( ).) The part will be included in the cost calculation spreadsheet if any of its variants matches the `--variant` argument.

3.6.1 Old-Style Variants

KiCost supports another way of specifying the variant associated with a part. Using the example from above, labeling the part number for J1 as `kicost.v1:manf#` will assign it to the v1 variant. This method is not as flexible as using the `variant` field and may be removed in future versions of KiCost.

3.7 "Do Not Populate" Parts

Some parts in a schematic are not intended for insertion on the final board assembly. These "do not populate" (DNP) parts can be assigned a field called `DNP` or `NOPOP`. Setting the value of this field to a non-zero number or any string will cause this part to be omitted from the cost calculation spreadsheet.

3.8 Showing Extra Part Data in the Spreadsheet

Sometimes it is desirable to show additional data for the parts in the spreadsheet. To do this, use the `--fields` command-line option followed by the names of the additional part fields you want displayed in the global data section of the of the spreadsheet:

```
kicost -i schematic.xml --fields fld1 fld2
```

3.9 Visual Cues in the Spreadsheet

In addition to the part cost information, the spreadsheet output by KiCost provides additional cues:

1. The `Qty` cell is colored to show the availability of a given part:
   - Red if the part is unavailable at any of the distributors.
   - Orange if the part is available, but not in sufficient quantity.
   - Yellow if there is enough of the part available, but not enough has been ordered.
   - Gray if no manufacturer or distributor part number was found in the BOM file.
2. The `Avail` cell is colored to show the availability of a given part at a particular distributor:
   - Red if the part is unavailable.
   - Orange if there is not sufficient quantity of the part available.
3. The `Unit$` and `Ext$` in each distributor cell is colored green to indicate the lowest price found across all the distributors.

3.10 Parallel Web Scraping

KiCost spends most of its time scraping the part data from the distributor web sites. In order to speed this up, many of the web scrapes can be run in parallel. By default, KiCost uses 30 parallel processes to gather the part data. This can be too much for some computers, so you can decrease the load using the `--num_processes` command-line option with the number of processes you want to spawn:

```
kicost -i schematic.xml --num_processes 10
```

In addition, you can use the `--serial` command-line option to force KiCost into single-threaded operation. This is equivalent to using `--num_processes 1`. (If you encounter problems running KiCost on a Windows PC with Python 2, then using this command may help.)

Some distributor may block multiple accesses of their websites such as those made by KiCost when scraping part information. To workaround this, each new scrape can be delayed by a time interval using the `--throttling_delay` option. In the follow example, each scrape of a website is only initiated after waiting for 100 milliseconds:

```
kicost -i schematic.xml --num_processes 10 --throttling_delay 0.1
```

3.11 Selecting Distributors to Scrape

You can get the list of part distributors that KiCost scrapes for data like this:

```
kicost --show_dist_list
Distributor list: digikey farnell local_template mouser newark rs tme
```

Since you may not have access to some of the distributors in that list, you can restrict scraping from only a subset of them as follows:

```
kicost -i schem.xml --include digikey mouser
```

Or you can exclude some distributors and scrape the rest:

```
kicost -i schem.xml --exclude farnell newark
```

3.12 Command-Line Options

```
usage: kicost [-h] [-v] [-i FILE.XML [FILE.XML ...]] [-o [FILE.XLSX]]
              [-f NAME [NAME ...]] [-var VARIANT [VARIANT ...]] [-w] [-s] [-q]
              [-np [NUM_PROCESSES]] [-ign NAME [NAME ...]]
              [-grp NAME [NAME ...]] [-d [LEVEL]]
              [-eda {kicad,altium,csv} [{kicad,altium,csv} ...]]
              [--show_dist_list] [--show_eda_list] [--no_collapse]
              [-e DIST [DIST ...]] [--include DIST [DIST ...]] [--no_scrape]
              [-rt [NUM_RETRIES]] [--throttling_delay [DELAY]] [--user]

Build cost spreadsheet for a KiCAD project.

optional arguments:
  -h, --help            show this help message and exit
  -v, --version         show program's version number and exit
  -i FILE.XML [FILE.XML ...], --input FILE.XML [FILE.XML ...]
                        One or more schematic BOM XML files.
  -o [FILE.XLSX], --output [FILE.XLSX]
                        Generated cost spreadsheet.
  -f NAME [NAME ...], --fields NAME [NAME ...]
                        Specify the names of additional part fields to extract
                        and insert in the global data section of the
                        spreadsheet.
  -var VARIANT [VARIANT ...], --variant VARIANT [VARIANT ...]
                        schematic variant name filter.
```

(continues on next page)

(continued from previous page)

```
  -w, --overwrite       Allow overwriting of an existing spreadsheet.
  -s, --serial          Do web scraping of part data using a single process.
  -q, --quiet           Enable quiet mode with no warnings.
  -np [NUM_PROCESSES], --num_processes [NUM_PROCESSES]
                        Set the number of parallel processes used for web
                        scraping part data.
  -ign NAME [NAME ...], --ignore_fields NAME [NAME ...]
                        Declare part fields to ignore when reading the BoM
                        file.
  -grp NAME [NAME ...], --group_fields NAME [NAME ...]
                        Declare part fields to merge when grouping parts.
  -d [LEVEL], --debug [LEVEL]
                        Print debugging info. (Larger LEVEL means more info.)
  -eda {kicad,altium,csv} [{kicad,altium,csv} ...], --eda_tool {kicad,altium,csv} [
↪{kicad,altium,csv} ...]
                        Choose EDA tool from which the XML BOM file
                        originated, or use csv for .CSV files.
  --show_dist_list      Show list of distributors that can be scraped for cost
                        data, then exit.
  --show_eda_list       Show list of EDA tools whose files KiCost can read,
                        then exit.
  --no_collapse         Do not collapse the part references like C1,C2,C3 into
                        C1-C3 in the spreadsheet.
  -e DIST [DIST ...], --exclude DIST [DIST ...]
                        Excludes the given distributor(s) from the scraping
                        process.
  --include DIST [DIST ...]
                        Includes only the given distributor(s) in the scraping
                        process.
  --no_scrape           Create a spreadsheet without scraping part data from
                        distributor websites.
  -rt [NUM_RETRIES], --retries [NUM_RETRIES]
                        Specify the number of attempts to retrieve part data
                        from a website.
  --throttling_delay [DELAY]
                        Specify minimum delay (in seconds) between successive
                        accesses to a distributor's website.
  --currency [CURRENCY-LOCALE], '--locale' [CURRENCY-LOCALE]
                        Define the priority locale/country and currency on the
                        scrape. Use the ISO4217 for currency and ISO3166:2 for
                        country. Input e.g.: `US`, `USD`, `US-USD` or `EUR-US`.
                        Currency is priritized over the locale/country. If give
                        country with more than one currency, it will be chosen,
                        in the sequence, `USD`, `EUR` or alphabetical order.
                        Default: `USD`.
  --user                Start the user guide to run KiCost passing the file
                        parameter give by "--input", all others parameters are
                        ignored.
```

3.13 Adding KiCost to the Context Menu (Windows Only)

You can add KiCost to the Windows context menu so you can right-click on an XML file and generate the pricing spreadsheet. To do this:

1. Open the registry and find the `HKEY_CLASSES_ROOT => xmlfile => shell` key. Then add a `KiCost`

key to it and, under that, add a `command` key. The resulting hierarchy of keys will look like this:

```
HKEY_CLASSES_ROOT
        |
        +-- xmlfile
              |
              +-- shell
                    |
                    +-- KiCost
                          |
                          +-- command
```

2. Set the value of the command to:

```
path_to_kicost -w -i "%1"
```

For example, the command value I use is:

```
C:\winpython3\python-3.4.3\scripts\kicost -w -i "%1"
```

3. If you have the GUIDE dependences insalled, it could be used:

```
path_to_kicost --user -i "%1"
```

So, KiCost will use the last preferences setted on the GUI to scrape, including which distributors to use, currency and others definitions.

4. Close the registry. KiCost should now appear when you right-click on an XML file.

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/xesscorp/kicost/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “feature” is open to whoever wants to implement it.

4.1.4 Write Documentation

kicost could always use more documentation, whether as part of the official kicost docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/xesscorp/kicost/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *kicost* for local development.

1. Fork the *kicost* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/kicost.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv kicost
$ cd kicost/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 kicost tests
$ python setup.py test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, 3.3, and 3.4, and for PyPy. Check https://travis-ci.org/xesscorp/kicost/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_kicost
```

CHAPTER 5

Credits

5.1 Development Lead

- XESS Corporation <info@xess.com>

5.2 Contributors

- Oliver Martin: https://github.com/oliviermartin
- Timo Alho: https://github.com/timoalho
- Steven Johnson: https://github.com/stevenj
- Diorcet Yann: https://github.com/diorcety
- Giacinto Luigi Cerone https://github.com/glcerone
- Hildo Guillardi Júnior https://github.com/hildogjr
- Adam Heinrich https://github.com/adamheinrich

CHAPTER 6

History

6.1 0.1.44 (2018-04-??)

- Add Upverter CSV compatibility.
- Fixed Mouser "quote price" exception in the price tiers.
- Fixed wxPython exception import.
- Use the datasheet link information from KiCad and other EDAs, given by 'datasheet' field.
- Now automatically merge 'description' and other fields to create the groups.
- GUI save last position and size and others improvements.
- Display additional information from the web page distributors and use as comment in the `cat#` column (just implemented on DigiKey yet).
- Now is possible to specify country/currency to be priorized on the distributors scrapes (just implemented on DigiKey yet).

6.2 0.1.43 (2018-03-15)

- Fixed RS scrape module.
- Added `--no_scrape` option to create spreadsheets without information from distributor websites.
- Added `--no_collapse` option to prevent collapsing part references in the spreadsheet.
- Added `--throttling_delay` option to add delay between accesses to distributor websites.
- Added `--show_eda_list` option to display the list of EDA tools supported by KiCost.
- Added capability to read multiple BOM files and merge them into the spreadsheet.
- Added `--group_fields` option to ignore differences in fields of the components and group them.
- Fixed the not ungrouping issue when `manf#` equal `None`.

- CSV now accepts files from Proteus and Eagle EDA tools.
- Cleared up unused Python imports and better placed functions into files (spreasheet creation files are now in `spreadsheet.py`).
- Added a KiCost stamp version at the end of the spreadsheet and file information in the beginning, if they are not inside it.
- Fixed issues related to user visualization in the spreadsheet (added gray formatted conditioning and the "exclude desc and manf columns").
- Added "user errors" and software scape in the case of not recognized references characters given the message of how to solve.
- Support for multiple quantity for a single manufacture code (before just worked when using multiple/sub-parts).
- Fixed the Altium EDA module.
- Created a graphical user interface based on wxWidgets (the dependence is asked to be installed at the first use).
- Added the `--user` option allow to use just `kicost --user -i %file` and others parameters will be got by the last configuration in the graphical interface (that save the user configurations).
- Added automatic recognition of the files of each EDA tool (for the graphical interface).

6.3 0.1.42 (2017-12-07)

- Processing of CSV files containing part information is now supported.
- Added `show_dist_list` option to display the list of distributors from which part cost data is available.
- Added capability to process multiple XML and CSV files.

6.4 0.1.41 (2017-11-16)

- Fixed exception caused by missing 'href' key in product links extracted by TME module.

6.5 0.1.40 (2017-11-02)

- Fixed exceptions caused by .xml files without a title block or part library section.

6.6 0.1.39 (2017-10-10)

- Part number separator characters can now be escaped with backslashes in case they are actually part of part numbers.

6.7 0.1.38 (2017-10-09)

- Fixed webscrape retry error in TME distributor module.

6.8 0.1.37 (2017-10-09)

- A part manf# field can now contain multiple subpart numbers. Each part number can be assigned a multiplier to indicate the quantity of the subpart needed for each part.
- Unit price cells for parts now show complete Qty/Price table as a cell comment.
- Part quantity cells are now color-coded to indicate parts with insufficient availability.
- Part quantity cells are now color-coded to indicate parts for which insufficient quantity has been ordered.
- Project name, company, and date are now shown in the spreadsheet.
- New distributor can now be added just by creating a submodule in `distributors`.
- Added distributor TME.
- Added `--retries` option to set the number of attempts at loading a distributor webpage.
- Fixed problem where "kicost:dnp" field was not recognized.

6.9 0.1.36 (2017-08-14)

- Parts may now be assigned to a variant by giving them a `variant` field.
- Parts may now be assigned to multiple variants.
- Parts may be designated as "do not populate" by giving them a `DNP` field.
- DNP parts or parts not in the current variant will not appear in the cost spreadsheet.

6.10 0.1.35 (2017-04-24)

- Fixed bug in scraping RS website when a part search results in a list of matches instead of a single product page.

6.11 0.1.34 (2017-03-31)

- Fixed crash caused by uninitialized array in Digikey webscraping module.
- Place any available scraped part info into spreadsheet even if part is not available from a distributor.
- Removed unused imports from distributor modules.

6.12 0.1.33 (2017-02-23)

- Surround worksheet name with quotes in case it contains spreadsheet operators.
- Fixed extraction of product links from Farnell product tables.

6.13 0.1.32 (2017-02-14)

- Added options for including or excluding distributors.
- Updated web scrapers for various distributors.
- Added more debugging/logger statements.
- Updated some of the package requirements.

6.14 0.1.31 (2016-11-14)

- Giacinto Luigi Cerone added support for distributors Farnell and RS.

6.15 0.1.30 (2016-11-07)

- Manufacturer's part number field can now be labeled as 'manf#', 'mpn', 'pn', '#', etc. (See documentation.)
- Manufacturer field can now be labeled as 'manf' or 'manufacturer'.
- Distributor part number fields can now be labeled as 'digikey#', 'digikeypn', digikey_pn', 'digikey-pn', etc.

6.16 0.1.29 (2016-08-27)

- KiCost no longer fails if the <libparts>...</libparts> section is missing from the XML file.
- Documentation moved to Github Pages.

6.17 0.1.28 (2016-08-18)

- Fixed scraping of Digi-Key pages to correctly detect reeled parts and scrape alternate packaging options.

6.18 0.1.27 (2016-07-26)

- Fixed scraping of Digi-Key pages to correctly extract available quantity of parts.

6.19 0.1.26 (2016-07-25)

- Progress bar is explicitly deleted to prevent an error from occurring when the program terminates.

6.20 0.1.25 (2016-06-12)

- Contents of "Desc" field in component/library were being ignored when generating spreadsheet.

6.21 0.1.24 (2016-05-28)

- Fixed part scraping from Newark website.

6.22 0.1.23 (2016-04-12)

- Added progress bar.
- Added quiet option to suppress warning messages.
- 'manf#' and 'manf' fields are now both propagated to similar parts.

6.23 0.1.22 (2016-04-08)

- Extra part data can now be shown in the global data section of the spreadsheet by using the new `--fields` command-line option. This commit implements issue #8.

6.24 0.1.21 (2016-03-20)

- Parts with valid Digi-Key web pages were not appearing in the spreadsheet because they had strange quantity listings (e.g., input fields or 'call for quantities'. This commit fixes #36.

6.25 0.1.20 (2016-03-20)

- Prices of $0.00 were appearing in the spreadsheet for parts that were listed but not stocked. Parts having no pricing list no longer list a price in the sheet.
- Parts with short manf. numbers (e.g. 5010) were not found correctly in the distributor websites. The manufacturer name was added to the search string to increase the probability of the search finding the correct part.

6.26 0.1.19 (2016-02-12)

- Local parts weren't showing up in spreadsheet because of previous fix to omit parts that had no quantity field (non-stocked; not even 0). Fixed.

6.27 0.1.18 (2016-02-10)

- Made change to adapt to change in Digi-Key's part quantity field of their webpages.
- Omit parts from the spreadsheet that are listed but not stocked at a distributor.

6.28 0.1.17 (2016-02-09)

- Made changes to adapt to changes in Digi-Key's webpage format.

6.29 0.1.16 (2016-01-26)

- Added `--variant` command-line option for costing different variants of a single schematic.
- Added `--num_processes` command-line option for setting the number of parallel processes used to scrape part data from the distributor web sites.
- Added `--ignore_fields` command-line option for ignoring benign fields that might prevent identical parts from being grouped together.

6.30 0.1.15 (2016-01-10)

- Fixed exception caused when indexing with 'manf#' on components that didn't have that field defined.
- Replaced custom debug_print() with logging module.

6.31 0.1.14 (2015-12-31)

- When scraping a Digi-Key product list page, use both the manfufacturer's AND Digi-Key's number to select the closest match to the part number.

6.32 0.1.13 (2015-12-29)

- 'kicost:' can be prepended to schematic field labels to distinguish them from other app fields.
- Custom prices and documentation links can now be added to parts in the schematic.
- Web-scraping for part data is sped up using parallel processes.

6.33 0.1.12 (2015-12-03)

- Following the IP address mouser with redirect you to the nearest locale match, so the price will be in Euro if you are in Europe and the price decimal can be a comma.

6.34 0.1.11 (2015-12-02)

- Changed BOARD_COST field to UNIT_COST.
- Changed formatting of UNIT_COST field to make use monetary units.
- Changed format of debug messages.

6.35 0.1.10 (2015-10-08)

- Pushed lxml requirement back to 3.3.3 so linux mint would have fewer problems trying to install.

6.36 0.1.9 (2015-09-26)

- Fixed exception caused by Digi-Key part with 'call' as an entry in a part's price list.
- Fixed extraction of part quantities in Mouser web pages.
- Added randomly-selected user-agent strings so sites might be less likely to block scraping.
- Added ghost.py code for getting around Javascript challenge pages (currently inactive).

6.37 0.1.8 (2015-09-17)

- Added missing requirements for future and lxml packages.

6.38 0.1.7 (2015-08-26)

- KiCost now runs under both Python 2.7.6 and 3.4.

6.39 0.1.6 (2015-08-26)

- Mouser changed their HTML page format, so I changed their web scraper.

6.40 0.1.5 (2015-07-25)

- Corrected entrypoint in __main__.py.

6.41 0.1.4 (2015-07-09)

- Added conditional formatting to indicate which distributor had the best price for a particular part.
- Fixed calc of min unit price so it wouldn't be affected if part rows were sorted.

6.42 0.1.3 (2015-07-07)

- Added global part columns that show minimum unit and extended prices for all parts across all distributors.

6.43 0.1.2 (2015-07-04)

- Refactoring.
- To reduce the effort in adding manufacturer's part numbers to a schematic, one will now be assigned to a part if:
  1. It doesn't have one.
  2. It is identical to another part or parts which do have a manf. part number.

3. There are no other identical parts with a different manf. part number than the ones in item #2.

6.44 0.1.1 (2015-07-02)

- Fixed delimiter for Mouser online order cut-and-paste.

6.45 0.1.0 (2015-06-30)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search