

VSCODE-EPICS, A VSCODE EXTENSION TO ENLIGHTEN YOUR EPICS CODE

Victor Nadot†, Alexis Gaget, Francis Gohier, Françoise Gougnaud, Paul Lotrus, Stéphane Tzvetkov, *IRFU, CEA, Université Paris-Saclay, F-91191 Gif-sur-Yvette, France

Abstract

vscod-epics is a Visual Studio Code extension developed by CEA Irfu that aims to enlighten your EPICS code. The development was originally initiated by one of our students in 2017.

This module makes developer life easier, improves code quality and helps to standardize EPICS code.

It provides syntax highlighting, snippets and header template for EPICS files and snippets for WeTest[1].

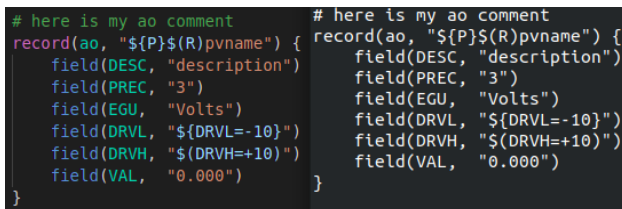
vscod-epics module is based on Visual Studio Code language Extension and it uses basic JSON files that make feature addition easy.

The number of downloads increases, version after version, and the various feedbacks we receive motivate us to strongly maintain it for the EPICS community. Since 2019, several other laboratories of the EPICS community have participated in the improvement of the module and it seems to have a nice future (linter, snippet improvements, specific language support, ...).

The module is available for free on Visual Studio Code marketplace[2] and on EPICS extension GitHub[3]. CEA Irfu is open to bug notifications, enhancement suggestions and merge requests, in order to continuously improve vscod-epics.

FEATURES

The syntax highlighting (Figure 1) provides the functionality to detect the EPICS keywords for a large scope of EPICS files (“.db”, “.template”, “.substitutions”, “.cmd”, “.st”, “.proto”, “.c”). As a concrete example, record types, record fields, macros and comments are supported for “.db” files.



```
# here is my ao comment # here is my ao comment
record(ao, "${P}${R}pvname") { record(ao, "${P}${R}pvname") {
  field(DESC, "description")   field(DESC, "description")
  field(PREC, "3")             field(PREC, "3")
  field(EGU, "Volts")          field(EGU, "Volts")
  field(DRVL, "${DRVL=-10}")   field(DRVL, "${DRVL=-10}")
  field(DRVH, "${DRVH=+10}")   field(DRVH, "${DRVH=+10}")
  field(VAL, "0.000")          field(VAL, "0.000")
}
```

Figure 1: With and without syntax highlighting.

Snippets are yet an even more powerful feature of the extension (see Figure 2). It makes it possible to generate any records with the basic EPICS fields. They are very valuable for new EPICS developers.

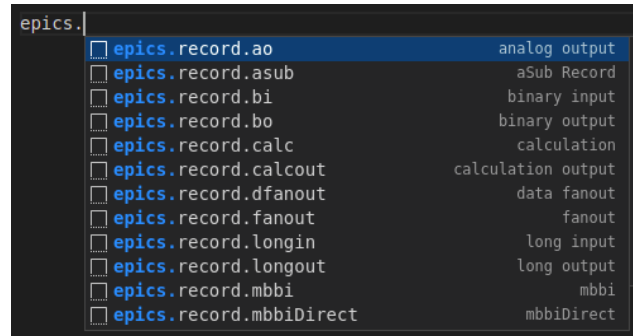
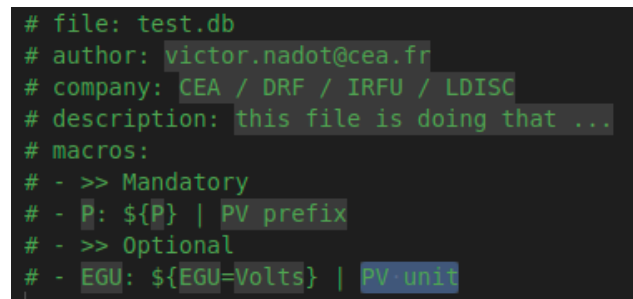


Figure 2: Scrollable list of available snippets.

Another interesting feature is the header template generation (see Figure 3). Largely inspired from our collaboration with ESS, the headers are composed of the following fields:

1. “file”: the file name is automatically written at the top of the header. This is useful when you use substitution files that generate a large “.db” file with all your templates inside. Thus, you can easily see the different template demarcations in the generated “.db” file.
2. “author”: the author and company references.
3. “description”: a short presentation describing the file scope.
4. “macros” :in order to improve code readability, it is strongly advised to add the list of the macros used in the file with a short description. They are split in two types. Mandatory: they are required to be defined at higher level to use the template. Optional: they have default values so they are not required to be defined at higher level to use the template;



```
# file: test.db
# author: victor.nadot@cea.fr
# company: CEA / DRF / IRFU / LDISC
# description: this file is doing that ...
# macros:
# - >> Mandatory
# - P: ${P} | PV prefix
# - >> Optional
# - EGU: ${EGU=Volts} | PV unit
```

Figure 3: File header provided by vscod-epics extension.

More snippet can be added as long as the language you are using is part of the EPICS extensions[4]. For instance WeTest, a software developed by CEA/Irfu to automate acceptance tests and shared with EPICS extension, has snippets integrated to epics-vscode module.

Besides making developer life easier, vscod-epics extension improves your code quality and code homogeneity

* † victor.nadot@cea.fr

Content from this work may be used under the terms of the CC BY 3.0 licence (© 2022). Any distribution of this work must maintain attribution to the author(s), title of the work, publisher, and DOI

across your developments. Indeed, the snippets feature will suggest the most used EPICS fields by record (they are not absolutely required but they help to improve the overall quality of your EPICS database). Here is a concrete example: an “ao” snippet would suggest to fill limits (DRVL/DRVH), unit (EGU), and precision (PREC). Note that for all snippets, the description (DESC) field is also suggested. All of these fields are not mandatory to make your EPICS database work, but it really improves it, especially from the client side: units can be displayed on GUI, GUI widgets can use PV limits, etc.

vscode-epics extension is based on JSON files in order to describe its different features. It makes it easy to read, to understand the implemented features and to add new ones.

EXTENSIONLIFE

The number of downloads increases quite linearly (see Figure 4), version after version, and the various feedbacks we receive motivate us to strongly maintain it for the EPICS community.

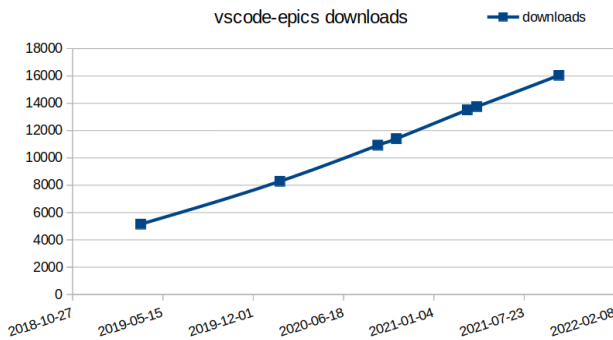


Figure 4: Vscode-epics extension downloads over the years.

Since 2019, some laboratories of the EPICS community have participated in the improvement of the module with significant contributions made by (see Figure 5):

- Institute of Plasma Physics (IPP CAS)
- European Spallation Source ERIC (ESS)
- Stanford Linear Accelerator Center (SLAC)
- Diamond Light Source (DLS)

It really is encouraging to receive some contributions from the EPICS community.

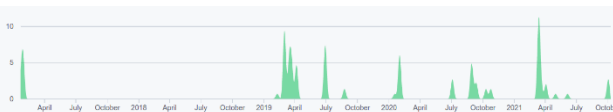


Figure 5: Contributions over the years.

The module is available on Visual Studio Code marketplace and on EPICS extension GitHub. CEA Irfu is open to bug notifications, enhancement suggestions and merge requests to continuously improve vscode-epics.

The current release is 1.1.0. The changelog (and the git history) is tracing the whole code history. In order to manage issues, I use the GitHub “Projects”. Its kanban provides a good view of the tickets (see Figure 6).

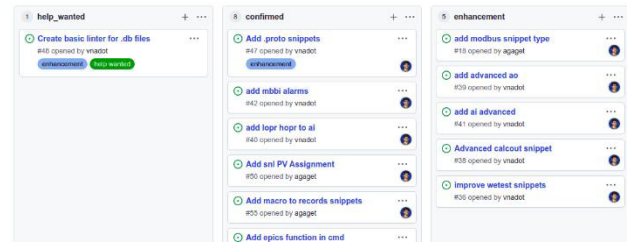


Figure 6: Vscode-epics kanban on Github.

FUTURE DEVELOPMENTS

There are few features in the pipelines, prioritized in two categories: confirmed and enhancements. There is also a “help wanted” column that contains the linter enhancement. The vscode-epics linter requires a bigger development so any help from the EPICS community, especially from someone with a linter development experience, would be appreciated. In addition, some automatic tests will be needed to avoid regressions.

Recently a colleague from Irfu CEA started a similar project for Vim (and NeoVim): epics-syntax.vim[5]. vscode-epics extension JSON files were a useful starting base. However, for now, only syntax highlighting is supported. In the longterm the same features as vscode-epics should be supported.

CONCLUSION

vscode-epics extension improves the EPICS code homogeneity and makes EPICS developers’ life easier thanks to its syntax highlighting, snippets and other features.

The extension has reached a stable state for several months now. The code is available on EPICS extension GitHub repository, it is open to merge requests and issue opening. GitHub kanban makes it easy to manage, and it provides the EPICS community with a good overview of the current project status.

REFERENCES

- [1] WeTest, <https://github.com/epics-extensions/WeTest>
- [2] vscode-epics extension, <https://marketplace.visualstudio.com/items?itemName=nsd.vscode-epics>
- [3] vscode-epics code, <https://github.com/epics-extensions/vscode-epics>
- [4] EPICS extensions on github, <https://github.com/epics-extensions>
- [5] epics-syntax.vim, <https://github.com/minijackson/epics-syntax.vim>