tree-sitter-epics

EPICS languages parsers for tool building

Gabriel Desmarchelier & Rémi Nicole

2026-04-21

1. EPICS tools

Depict : Document EPICs daTabase

- “Please provide documentation including a list of all variables.”
- As developers we like to automatically rebuild our documentation on any change.
- That’s what Depict allows you to do !
- Just add the variable description as a comment above the record.
- And create your Markdown PV table from it.

```
# RF injection ratio calculation
record(calcout,"${P}RfInjRatioCalc"){
    field(INPA,"${P}SyncAvgUCiAmpMes CP")
    field(INPB,"${P}SyncAvgUAmpAmpMes CP")
    field(CALC,"A/B")
    field(OUT,"${P}RfInjRatioMes PP")
}

# RF injection ratio measurement (UCi/UAmp)
record(ai,"${P}RfInjRatioMes"){
    field(PREC,3)
}

# RF injection ratio alarm low threshold
record(ao,"${P}RfInjRatioThrSet"){
    field(PREC,3)
    field(DRVL,0)
    field(DRVH,1)
}

# Minimal value of UAmp for injection ratio to raise an alarm
record(ao,"${P}UAmpThrSet"){
    field(VAL,0.5)
    field(PREC,3)
    field(EGU,"mV")
}

# RF injection alarm calculation
record(calcout,"${P}RfInjStatCalc"){
    field(INPA,"${P}RfInjRatioCalc CP")
    field(INPB,"${P}RfInjRatioThrSet CP")
    field(INPC,"${P}SyncAvgUAmpAmpMes CP")
    field(INPD,"${P}UAmpThrSet CP")
    field(CALC,"C<D?0:A>B?1:2")
    field(OUT,"${P}RfInjStat PP")
}

# RF injection alarm status
record(mbbi,"${P}RfInjStat"){
    field(ZRST,"RF TOO LOW")
    field(ONST,"RF INJ OK")
    field(TWST,"RF INJ ALARM")
}

# Forwarding injection alarm to sequence error
record(calcout,"${P}RfInjAlarmFanout"){
    field(INPA, "${P}RfInjStat CP")
    field(INPB, "${P}CavErrCode CP")
    field(CALC, "A==2?2:B")
    field(OUT,"${P}CavErrCode PP")
}
```

cavControl_RfInjection.template

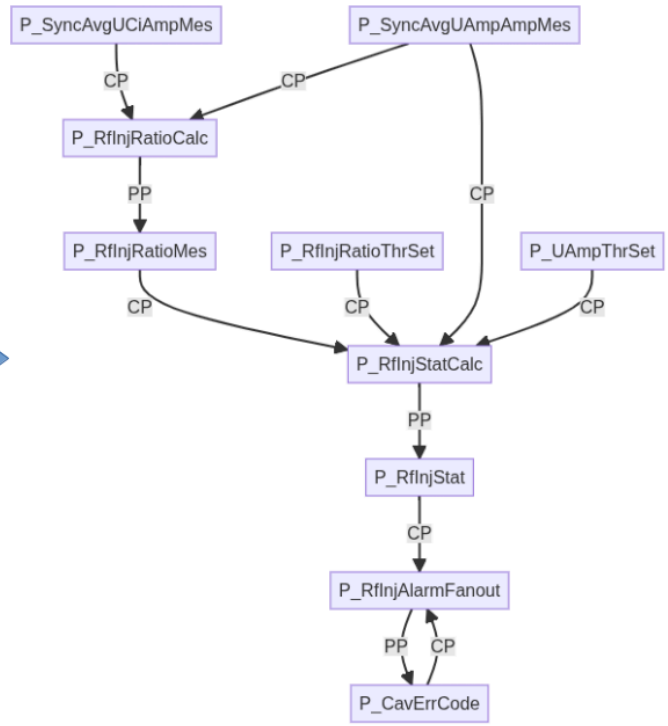
Name	Type	EGU	Description
\${P}RfInjRatioCalc	calcout		RF injection ratio calculation
\${P}RfInjRatioMes	ai		RF injection ratio measurement (UCi/UAmp)
\${P}RfInjRatioThrSet	ao		RF injection ratio alarm low threshold
\${P}UAmpThrSet	ao	mV	Minimal value of UAmp for injection ratio to raise an alarm
\${P}RfInjStatCalc	calcout		RF injection alarm calculation
\${P}RfInjStat	mbbi		RF injection alarm status
\${P}RfInjAlarmFanout	calcout		Forwarding injection alarm to sequence error

Dryed : DRaw Your Epics Database

- “Where on earth was this PV written from ?”
- Just run Dryed to have an overview of your EPICS database process links !
- Based on the Mermaid diagramming tool.
- Still an Alpha version (need to work on macro resolution to provide overiew of complex databases).

```
# RF injection ratio calculation
record(calcout,"${P}RfInjRatioCalc"){
    field(INPA,"${P}SyncAvgUCiAmpMes CP")
    field(INPB,"${P}SyncAvgUAmpAmpMes CP")
    field(CALC,"A/B")
    field(OUT,"${P}RfInjRatioMes PP")
}

# RF injection ratio measurement (UCi/UAmp)
record(ai,"${P}RfInjRatioMes"){
    field(PREC,3)
}

# RF injection ratio alarm low threshold
record(ao,"${P}RfInjRatioThrSet"){
    field(PREC,3)
    field(DRVL,0)
    field(DRVH,1)
}

# Minimal value of UAmp for injection ratio to raise an alarm
record(ao,"${P}UAmpThrSet"){
    field(VAL,0.5)
    field(PREC,3)
    field(EGU,"mV")
}

# RF injection alarm calculation
record(calcout,"${P}RfInjStatCalc"){
    field(INPA,"${P}RfInjRatioCalc CP")
    field(INPB,"${P}RfInjRatioThrSet CP")
    field(INPC,"${P}SyncAvgUAmpAmpMes CP")
    field(INPD,"${P}UAmpThrSet CP")
    field(CALC,"C<D?0:A>B?1:2")
    field(OUT,"${P}RfInjStat PP")
}

# RF injection alarm status
record(mbbi,"${P}RfInjStat"){
    field(ZRST,"RF TOO LOW")
    field(ONST,"RF INJ OK")
    field(TWST,"RF INJ ALARM")
}

# Forwarding injection alarm to sequence error
record(calcout,"${P}RfInjAlarmFanout"){
    field(INPA, "${P}RfInjStat CP")
    field(INPB, "${P}CavErrCode CP")
    field(CALC, "A==2?2:B")
    field(OUT,"${P}CavErrCode PP")
}
```

```
flowchart
    P_SyncAvgUCiAmpMes--CP-->P_RfInjRatioCalc
    P_SyncAvgUAmpAmpMes--CP-->P_RfInjRatioCalc
    P_RfInjRatioCalc--PP-->P_RfInjRatioMes
    P_RfInjRatioMes--CP-->P_RfInjStatCalc
    P_RfInjRatioThrSet--CP-->P_RfInjStatCalc
    P_SyncAvgUAmpAmpMes--CP-->P_RfInjStatCalc
    P_UAmpThrSet--CP-->P_RfInjStatCalc
    P_RfInjStatCalc--PP-->P_RfInjStat
    P_RfInjStat--CP-->P_RfInjAlarmFanout
    P_CavErrCode--CP-->P_RfInjAlarmFanout
    P_RfInjAlarmFanout--PP-->P_CavErrCode
```

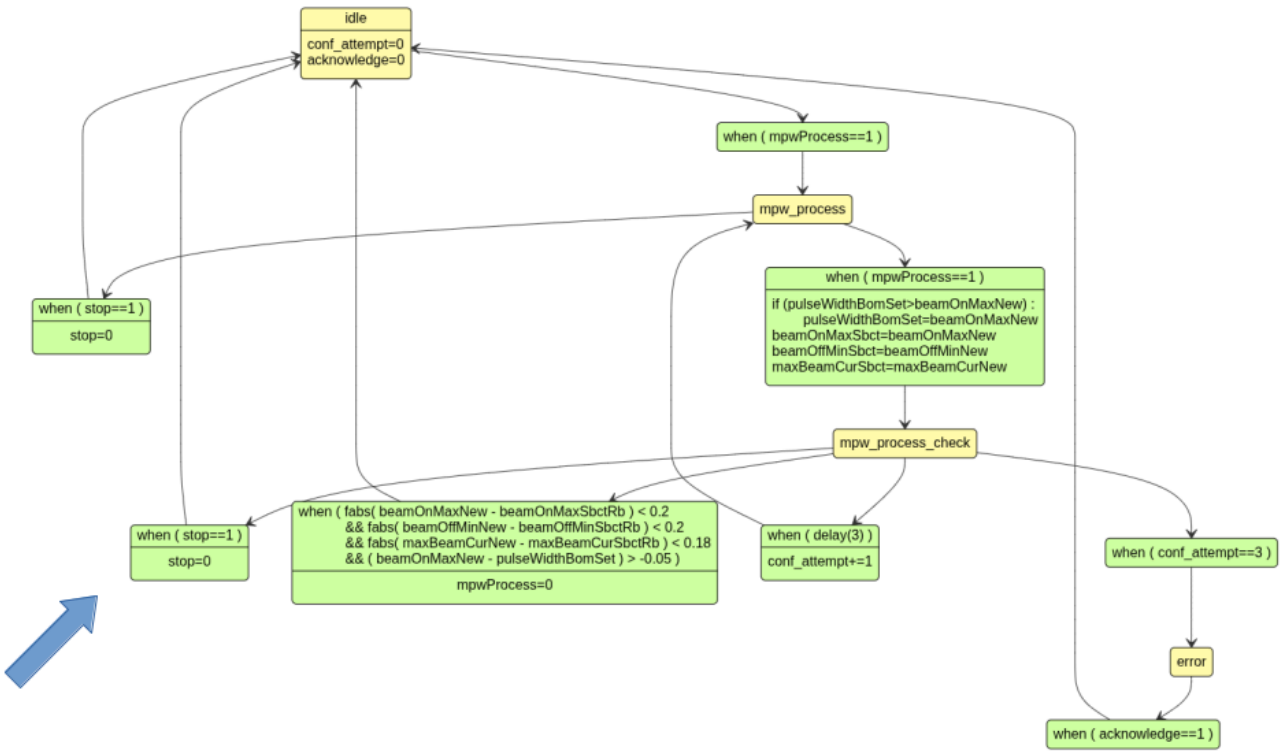


Plasma : PLAin State Machine Acquaintance

- Do we want to find an acronym at any price ? –> YES
- “Well, no one can really tell what’s happening in that SNL state machine…”
- Plasma builds diagrams of you SNL program.
- Based on the Mermaid diagramming tool.
- Fancy for small codes, messy for big ones : we’re missing a better text-described diagramming tool.

```
            acknowledge=0;
            pvPut(acknowledge);
        }

        when(mpwProcess==1) {
        } state mpw_process

    }

    state mpw_process
    {
        when(stop==1) {
            stop=0;
            pvPut(stop);
        } state idle

        when(mpwProcess==1) {
            printLog(ssId, pvIndex(message),message, "Processing : applying configuration to BOM and SBCT");
            // prevent a too wide pulse to be applied in the BOM, in order to avoid shutbeam
            if(pulseWidthBomSet>beamOnMaxNew){
                pulseWidthBomSet=beamOnMaxNew;
            }
            pvPut(pulseWidthBomSet);
            // Set the protective BOOM parameters for operation
            beamOnMaxSbct=beamOnMaxNew;
            beamOffMinSbct=beamOffMinNew;
            maxBeamCurSbct=maxBeamCurNew;
            pvPut(beamOnMaxSbct);
            pvPut(beamOffMinSbct);
            pvPut(maxBeamCurSbct);
        } state mpw_process_check
```

```
---
title: state_set_1
---
stateDiagram
    classDef state_style fill:#FFFAAA,stroke:black,color:black
    classDef transition_style fill:#CFFFA0,stroke:black,color:black
    idle:::state_style
    idle : idle
    idle:  conf_attempt=0
    idle:  acknowledge=0
    idle_transition_1:::transition_style
    idle --> idle_transition_1
    idle_transition_1 : when (  mpwProcess==1  )
    idle_transition_1 --> mpw_process
    mpw_process:::state_style
    mpw_process : mpw_process
    mpw_process_transition_1:::transition_style
    mpw_process --> mpw_process_transition_1
    mpw_process_transition_1 : when (  stop==1  )
    mpw_process_transition_1:  stop=0
    mpw_process_transition_1 --> idle
    mpw_process_transition_2:::transition_style
    mpw_process --> mpw_process_transition_2
```



But what do all these tools have in common ?

- Drum roll please…

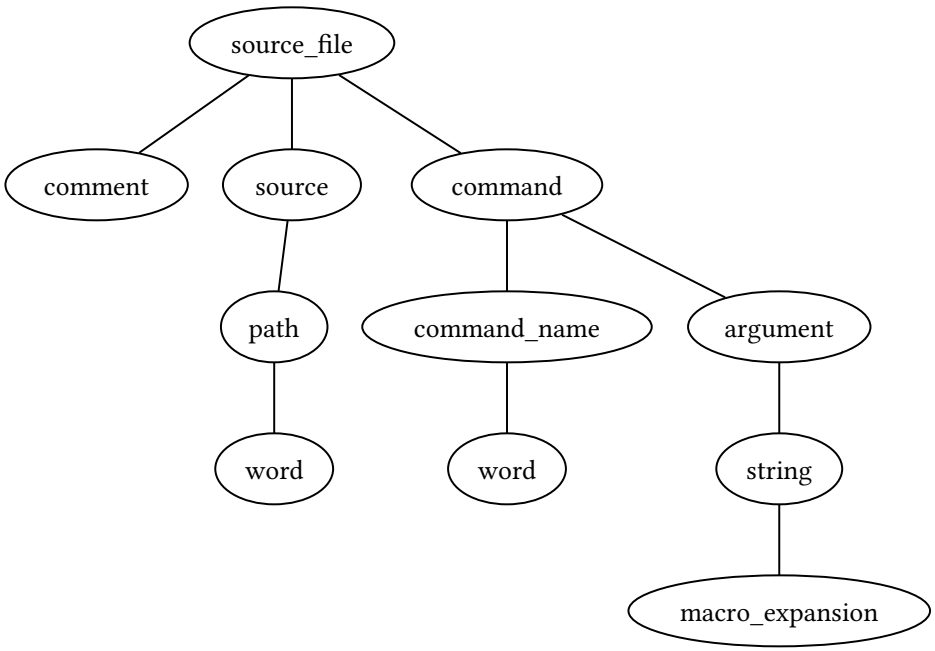
2. tree-sitter grammars

Goal of a tree-sitter parser

Example `.cmd` file

```
# This is a comment
< envPaths

cd "${TOP}"
```

→



Developed parsers

tree-sitter-epics-db	for `.db` & `.dbd` files
tree-sitter-epics-cmd	for `.cmd` files
tree-sitter-epics-msi-substitution	for `.substitutions` files
tree-sitter-epics-msi-template	for macros inside files above
tree-sitter-streamdevice-proto	for StreamDevice `.proto` files
tree-sitter-snl	for the Sequencer's `.st` & `.stt` files

How to use the parsers

Use the appropriate Python, Rust, or JavaScript package.

How to use the parsers

Use the appropriate Python, Rust, or JavaScript package.

Python example

```python
import tree_sitter_snl
from tree_sitter import Language, Parser
from pathlib import Path

parser = Parser(Language(tree_sitter_snl.language()))
tree = parser.parse(Path("myFile.st").read_bytes())
```

How to use the parsers — II

Then, either:

- Walk the syntax tree manually yourself
- Use tree-sitter queries

tree-sitter queries

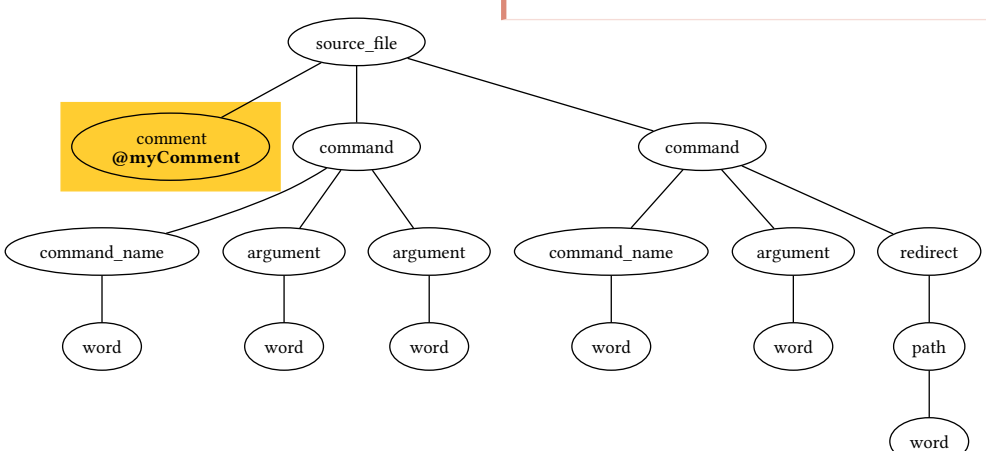
Match a certain “query” or “pattern” that happen in a tree.

Example `cmd` query

```
(comment) @myComment
```

Example `cmd` file

```
# This is a comment
command(arg1, arg2)
echo hello >> myFile.txt
```



tree-sitter queries

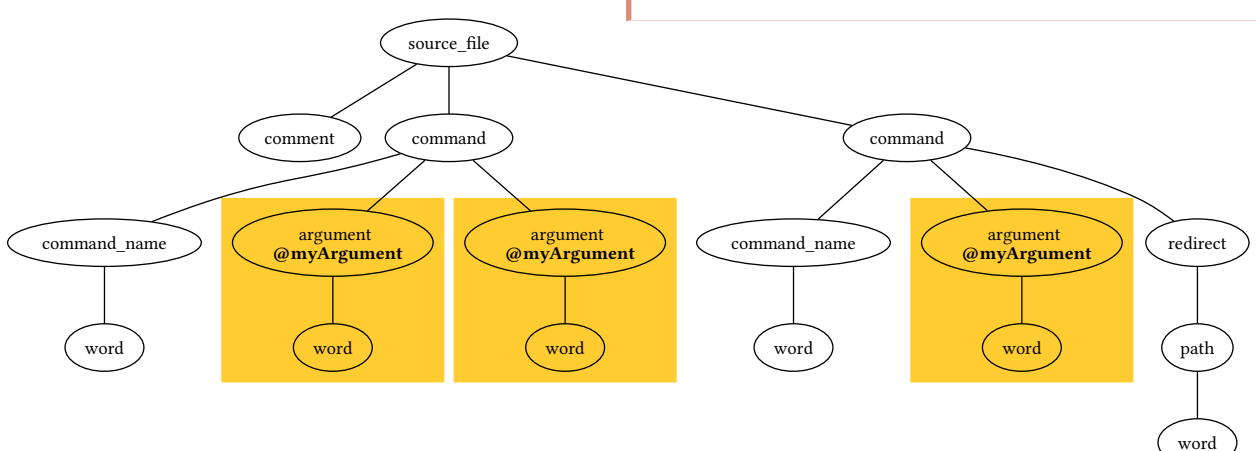
Match a certain “query” or “pattern” that happen in a tree.

Example `cmd` query

```
(argument) @myArgument
```

Example `cmd` file

```
# This is a comment
command(arg1, arg2)
echo hello >> myFile.txt
```



tree-sitter queries

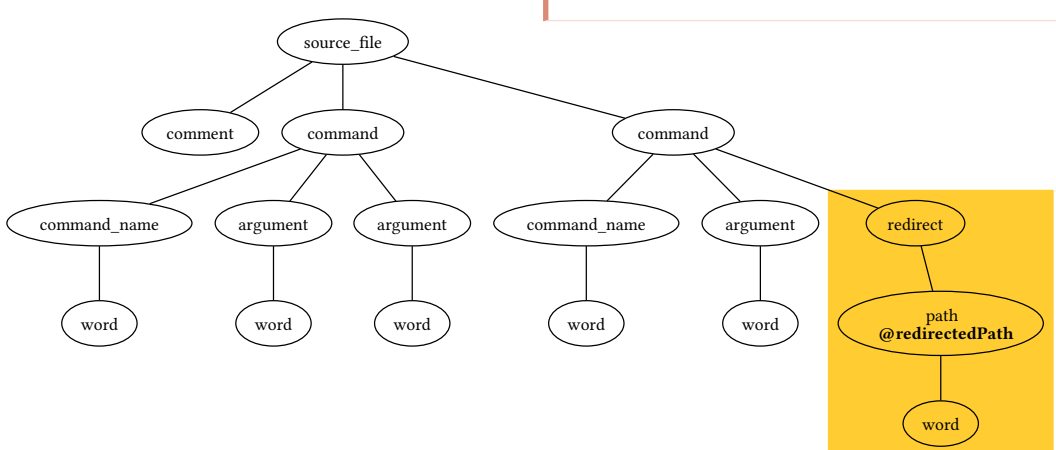
Match a certain “query” or “pattern” that happen in a tree.

Example `cmd` query

```
(redirect
  (path) @redirectedPath)
```

Example `cmd` file

```
# This is a comment
command(arg1, arg2)
echo hello >> myFile.txt
```



More complex queries

Example `db` query

```
(
  (comment)* @record_documentation
```

More complex queries

Example `db` query

```
(
  (comment)* @record_documentation
  .
  (record_instance
```

More complex queries

Example `db` query

```
(
  (comment)* @record_documentation
  .
  (record_instance
    name: (record_name) @record_name)
```

More complex queries

Example `db` query

```
(
  (comment)* @record_documentation
  .
  (record_instance
    name: (record_name) @record_name)
)
```

Example matched `db` code

```
# hello    ← @record_documentation
# world    ← @record_documentation
record(lsi, "$(user):example:version") {
    #       ↑↑↑↑↑ @record_name ↑↑↑↑↑↑
    field(DTYP, "example version")
    ...
}

#          ↓↓↓ @record_name ↓↓↓
record(ai, "$(user):example:ai") {}
```

One more db query example

```
[
  (record_instance)
  (include_statement)
  ; ...
] @prepend_hardline @allow_blank_line_before
```

One more db query example

```
[
  (record_instance)
  (include_statement)
  ; ...
] @prepend_hardline @allow_blank_line_before

(record_instance "," @append_spaced_softline)
```

One more db query example

```
[
  (record_instance)
  (include_statement)
  ; ...
] @prepend_hardline @allow_blank_line_before

(record_instance "," @append_spaced_softline)

(record_instance
  "{" @prepend_space @append_spaced_softline @append_indent_start
  "}" @prepend_spaced_softline @prepend_indent_end)
```

One more db query example

```
[
  (record_instance)
  (include_statement)
  ; ...
] @prepend_hardline @allow_blank_line_before

(record_instance "," @append_spaced_softline)

(record_instance
  "{" @prepend_space @append_spaced_softline @append_indent_start
  "}" @prepend_spaced_softline @prepend_indent_end)
```

Surprise, we've got a formatter

Using the Topiary project, a formatter that uses tree-sitter grammars and queries.

Currently capable of formatting `.cmd`, `.db`, and `.dbd` files.

For `.cmd`, transforms: `a b c` into `a("b", "c")`

Surprise, we've got a formatter

Using the Topiary project, a formatter that uses tree-sitter grammars and queries.

Currently capable of formatting `.cmd`, `.db`, and `.dbd` files.

For `.cmd`, transforms: `a b c` into `a("b", "c")`

Currently unpublished / unfinished.

Limitations

tree-sitter makes parsers, not evaluators, it isn’t capable of evaluating macros.

Limitations

tree-sitter makes parsers, not evaluators, it isn't capable of evaluating macros.

Some valid code with macros can't be parsed by those tree-sitter grammar:

```
# Invalid for tree-sitter-epics-db
record(ai, ${HELLO}) { ... }

# But this is valid
record(ai, "${HELLO}") { ... }
```

Limitations

tree-sitter makes parsers, not evaluators, it isn't capable of evaluating macros.

Some valid code with macros can't be parsed by those tree-sitter grammar:

```
# Invalid for tree-sitter-epics-db
record(ai, ${HELLO}) { ... }

# But this is valid
record(ai, "${HELLO}") { ... }
```

Limitations

tree-sitter makes parsers, not evaluators, it isn't capable of evaluating macros.

Some valid code with macros can't be parsed by those tree-sitter grammar:

My advice:

- expand macros *before* parsing the files
- add quotes everywhere

Strictness

`tree-sitter-epics` grammars *may* also be more strict than epics-base.

Valid for epics-base, not for the tree-sitter parser

```
echo hello # say hello
#          ↑ missing "\n"
```

Valid for epics-base, not for the tree-sitter parser

```
echo(hello
#          ↑ missing ")"
```

Valid for epics-base, not for the tree-sitter parser

```
echo()))))hello ((())) ((, world
#     ↑↑↑↑↑↑↑↑↑ error ↑↑↑↑↑↑↑↑↑↑
```

Valid for epics-base, not for the tree-sitter parser

```
echo hello,,,,,,,,,,,,,,,, world
#          ↑↑↑↑ error ↑↑↑↑
```

Valid for epics-base, not for the tree-sitter parser

```
file "example.db" {
    pattern { one, two,,,,,,,,,,,,,,,, three }
    #                    ↑↑↑↑ error ↑↑↑↑
        { 1, 2, 3 }
}
```

Valid for epics-base, not for the tree-sitter parser

```
file "example.db" {
    pattern { one, two,,,,,,,,,,,,,,,, three }
    #                    ↑↑↑↑ error ↑↑↑↑
        { 1, 2, 3 }
}
```

Perfectly valid `cmd`

```
# EPICS, yay!
dbLoadRecords("db/test1.db")     ()(),
dbLoadRecords("db/test2.db")     ()(),
dbLoadRecords("db/test3.db")         ,
dbLoadRecords("db/test4.db")     ()(),
dbLoadRecords("db/test5.db")     ()(),
dbLoadRecords("db/test6.db")         ,
dbLoadRecords("db/test7.db")    ,,,,,,
dbLoadRecords("db/test8.db")    ,
dbLoadRecords("db/test9.db")    ,()()
dbLoadRecords("db/test10.db")   ,()()
dbLoadRecords("db/test11.db")   ,
dbLoadRecords("db/test12.db")   ,()()
dbLoadRecords("db/test13.db")   ,()()
```