

Liquidshop 4

A look back & forth

Romain Beauxis, May 26, 2024

Le Programme

Le Programme

- General Updates

Le Programme

- General Updates
- What happened since liquidshop 3?

Le Programme

- General Updates
- What happened since liquidshop 3?
- Remaining liquidsoap 2.3.x roadmap

Le Programme

- General Updates
- What happened since liquidshop 3?
- Remaining liquidsoap 2.3.x roadmap
- Questions?

From the archive: Liquidsoap 2.3.x roadmap

- Update the liquidsoap book
- Rewrite the streaming loop
- Rewrite the clock implementation
- Take advantage of OCaml 5 concurrency
- Proper module support
- Address standard library performance issues
- Add developer tooling: linter/prettier, syntax highlighting, etc.

From the archive: Liquidsoap 2.3.x roadmap

- Update the liquidsoap book - **PENDING!**
- Rewrite the streaming loop
- Rewrite the clock implementation
- Take advantage of OCaml 5 concurrency
- Proper module support
- Address standard library performance issues
- Add developer tooling: linter/prettier, syntax highlighting, etc.

From the archive: Liquidsoap 2.3.x roadmap

- Update the liquidsoap book - **PENDING!**
- Rewrite the streaming loop - **DONE!** 🎉
- Rewrite the clock implementation
- Take advantage of OCaml 5 concurrency
- Proper module support
- Address standard library performance issues
- Add developer tooling: linter/prettier, syntax highlighting, etc.

From the archive: Liquidsoap 2.3.x roadmap

- Update the liquidsoap book - **PENDING!**
- Rewrite the streaming loop - **DONE!** 🎉
- Rewrite the clock implementation - **DONE!** 🎉
- Take advantage of OCaml 5 concurrency
- Proper module support
- Address standard library performance issues
- Add developer tooling: linter/prettier, syntax highlighting, etc.

From the archive: Liquidsoap 2.3.x roadmap

- Update the liquidsoap book - **PENDING!**
- Rewrite the streaming loop - **DONE!** 🎉
- Rewrite the clock implementation - **DONE!** 🎉
- Take advantage of OCaml 5 concurrency - **BLOCKED!** ⚠️
- Proper module support
- Address standard library performance issues
- Add developer tooling: linter/prettier, syntax highlighting, etc.

From the archive: Liquidsoap 2.3.x roadmap

- Update the liquidsoap book - **PENDING!**
- Rewrite the streaming loop - **DONE!** 🎉
- Rewrite the clock implementation - **DONE!** 🎉
- Take advantage of OCaml 5 concurrency - **BLOCKED!** ⚠️
- Proper module support - **PAUSED** ⏸️
- Address standard library performance issues
- Add developer tooling: linter/prettier, syntax highlighting, etc.

From the archive: Liquidsoap 2.3.x roadmap

- Update the liquidsoap book - **PENDING!**
- Rewrite the streaming loop - **DONE!** 🎉
- Rewrite the clock implementation - **DONE!** 🎉
- Take advantage of OCaml 5 concurrency - **BLOCKED!** ⚠️
- Proper module support - **PAUSED** ⏸️
- Address standard library performance issues - **ONGOING!** 😬
- Add developer tooling: linter/prettier, syntax highlighting, etc.

From the archive: Liquidsoap 2.3.x roadmap

- Update the liquidsoap book - **PENDING!**
- Rewrite the streaming loop - **DONE!** 🎉
- Rewrite the clock implementation - **DONE!** 🎉
- Take advantage of OCaml 5 concurrency - **BLOCKED!** ⚠️
- Proper module support - **PAUSED** ⏸️
- Address standard library performance issues - **ONGOING!** 😬
- Add developer tooling: linter/prettier, syntax highlighting, etc. - **DONE!** 🎉

Developer Tooling

Developer Tooling

- Prettier: formatting

Developer Tooling

- Prettier: formatting
- Tree-sitter/code-mirror: syntax highlighting

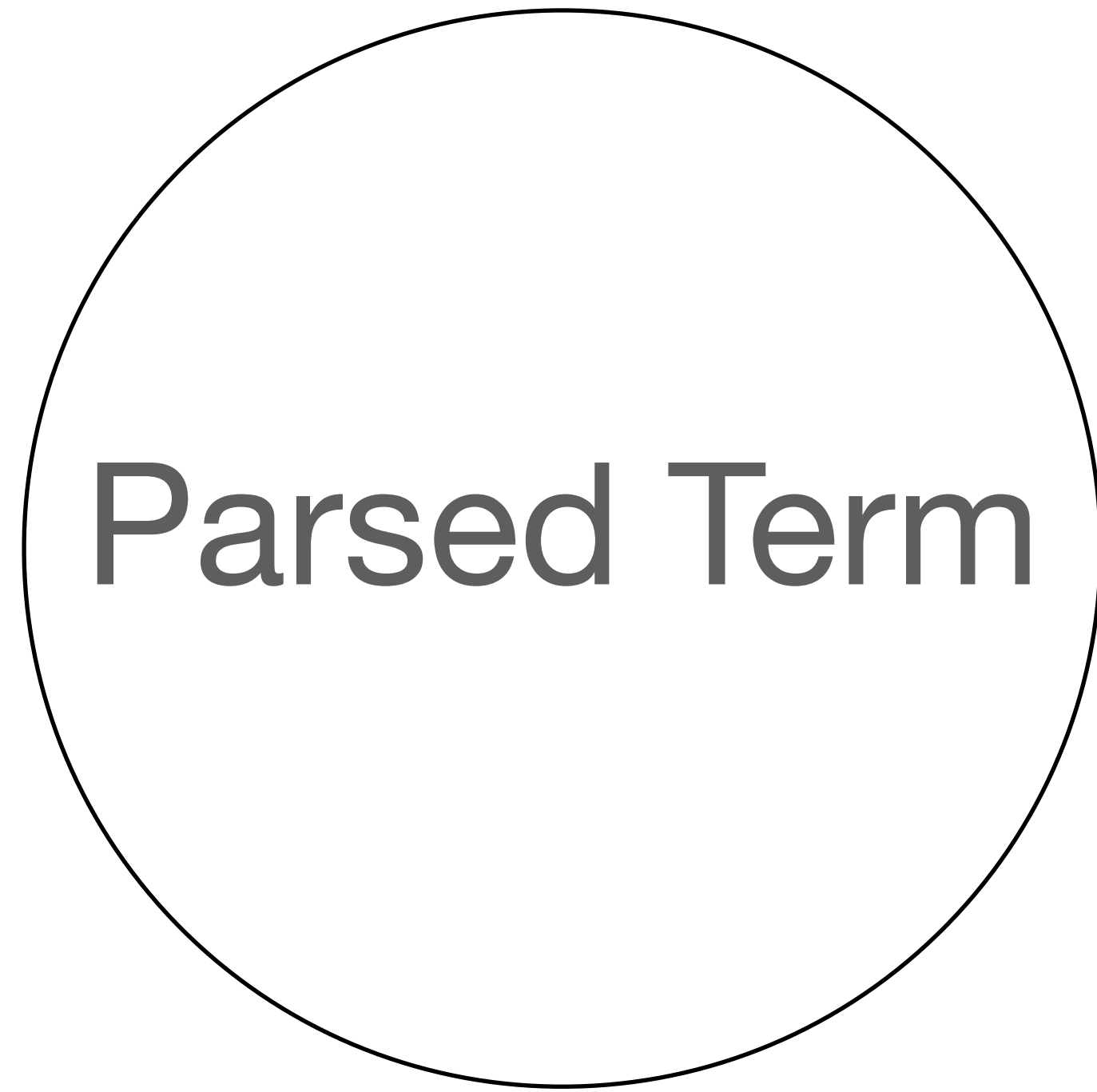
Developer Tooling

- Prettier: formatting
- Tree-sitter/code-mirror: syntax highlighting
- VSCode: syntax highlighting/formatting

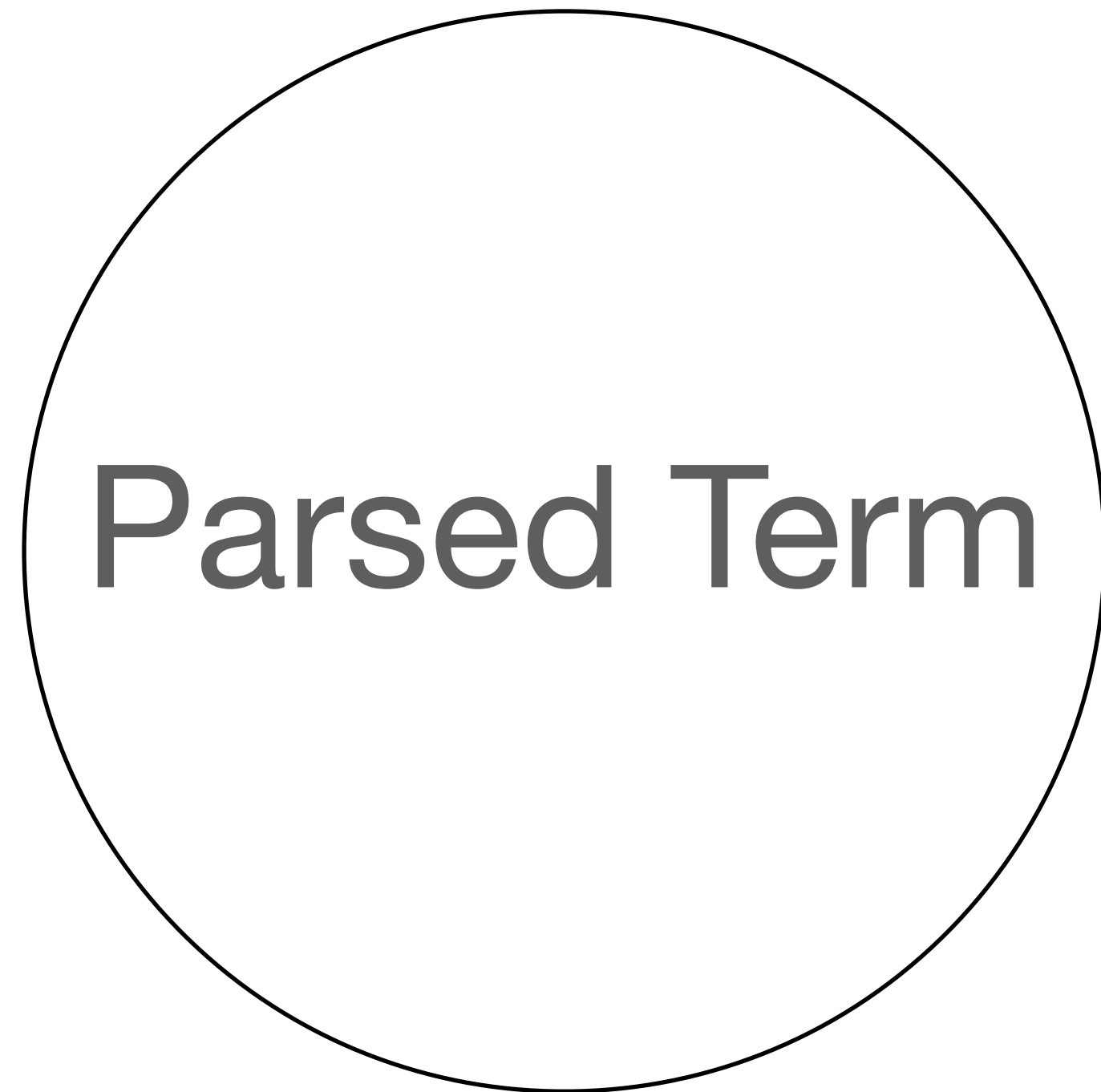
Developer Tooling

- Prettier: formatting
- Tree-sitter/code-mirror: syntax highlighting
- VSCode: syntax highlighting/formatting
- Github: syntax highlighting/code identification

Prettier



Prettier



```
list.add(x, 1)
```

Prettier

Parsed Term

```
list.add(x, 1)
```

```
`App  
  ( `Invoke {  
    invoked = `Var "list";  
    meth = "add"  
  },  
  [( "", `Var "x"); ( "", `Var "1") ] )
```

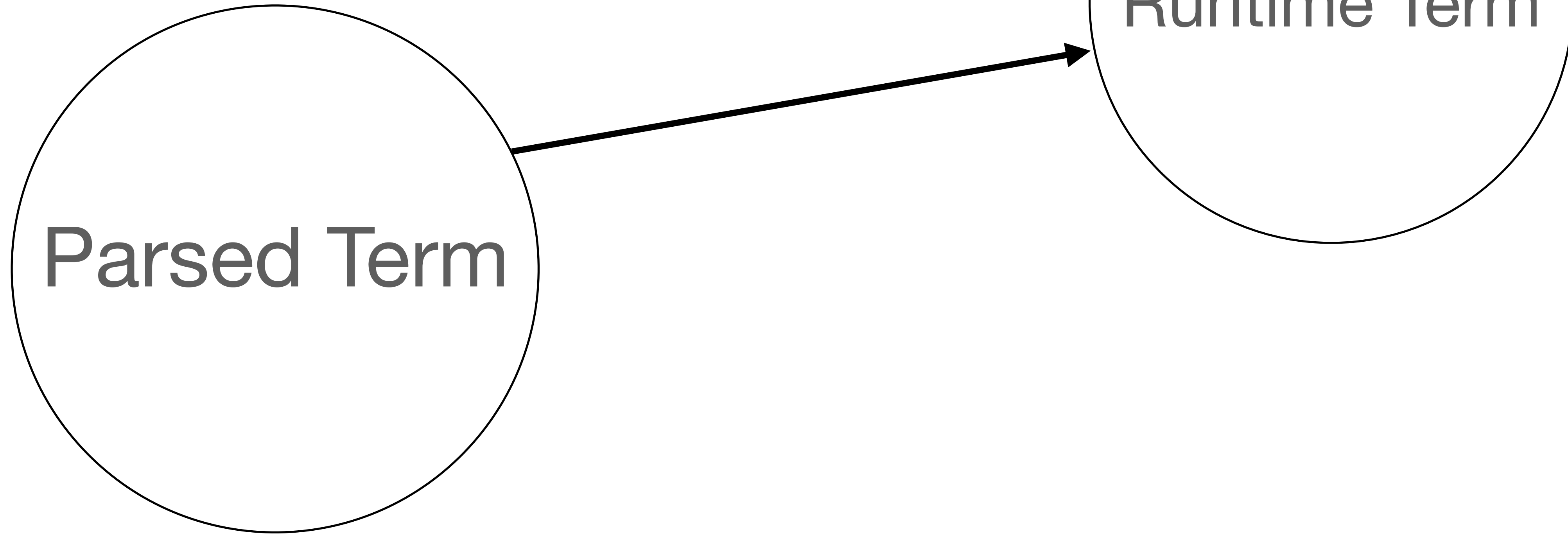
Prettier



Parsed Term

```
(* These terms are reduced at runtime *)
and parsed_ast =
  [ `If of _if
  | `Inline_if of _if
  | `If_def of if_def
  | `If_version of if_version
  | `If_encoder of if_encoder
  | `While of _while
  | `For of _for
  | `Iterable_for of iterable_for
  | `List of list_el list
  | `Try of _try
  | `Regex of string * char list
  | `Time_interval of time_el * time_el
  | `Time of time_el
  | `Def of _let * t
  | `Let of _let * t
  | `Binding of let * t
```

Prettier



Prettier

```
type 'a common_ast =  
  [ `Custom of custom_term  
  | `Tuple of 'a list  
  | `Null  
  | `Open of 'a * 'a  
  | `Var of string  
  | `Seq of 'a * 'a ]  
[@@deriving hash]
```



Runtime Term

```
type 'a runtime_ast =  
  [ `Int of int  
  | `Float of float  
  | `String of string  
  | `Bool of bool  
  | `Let of 'a let_t  
  | `List of 'a list  
  | `Cast of 'a * Type.t  
  | `App of 'a * (string * 'a) list  
  | `Invoke of 'a invoke  
  | `Encoder of 'a encoder  
  | `Fun of ('a, Type.t) func ]
```

Prettier



Runtime Term

```
| `String_interpolation (sep, l) ->
  let l =
    List.map
      (function
        | `String s -> `Term (mk_parsed ~pos (`String (sep, s)))
        | `Term tm ->
          `Term
            (mk_parsed ~pos
              (`App (mk_parsed ~pos (`Var "string"), [`Term ("", tm)]))))
      l
  in
  let op =
    mk_parsed ~pos
      (`Invoke
        {
          invoked = mk_parsed ~pos (`Var "string");
          meth = `String "concat";
          optional = false;
        })
  in
  to_ast ~env ~pos (`App (op, [`Term ("", mk_parsed ~pos (`List l))]))
```

Prettier

```
| `String_interpolation (sep, l) ->
  let l =
    List.map
      (function
        | `String s -> `Term (mk_parsed ~pos (`String (sep, s)))
        | `Term tm ->
          `Term
            (mk_parsed ~pos
              (`App (mk_parsed ~pos (`Var "string"), [`Term ("", tm)]))))
      l
  in
  let op =
    mk_parsed ~pos
      (`Invoke
        {
          invoked = mk_parsed ~pos (`Var "string");
          meth = `String "concat";
          optional = false;
        })
  in
  to_ast ~env ~pos (`App (op, [`Term ("", mk_parsed ~pos (`List l))]))
```



Runtime Term

```
"#{mpcdec} #{process.quote(f)} -"
```


Prettier

```
| `String_interpolation (sep, l) ->
  let l =
    List.map
      (function
        | `String s -> `Term (mk_parsed ~pos (`String (sep, s)))
        | `Term tm ->
          `Term
            (mk_parsed ~pos
              (`App (mk_parsed ~pos (`Var "string"), [`Term ("", tm)]))))
      l
  in
  let op =
    mk_parsed ~pos
      (`Invoke
        {
          invoked = mk_parsed ~pos (`Var "string");
          meth = `String "concat";
          optional = false;
        })
  in
  to_ast ~env ~pos (`App (op, [`Term ("", mk_parsed ~pos (`List l))]))
```

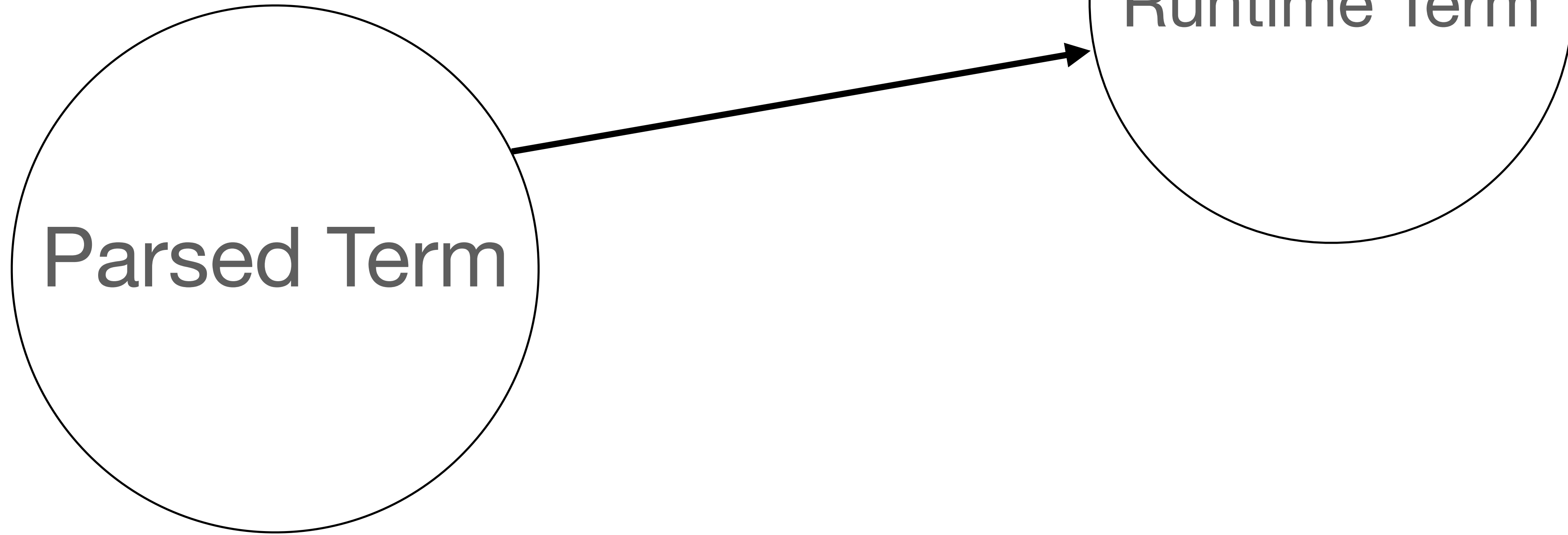


Runtime Term

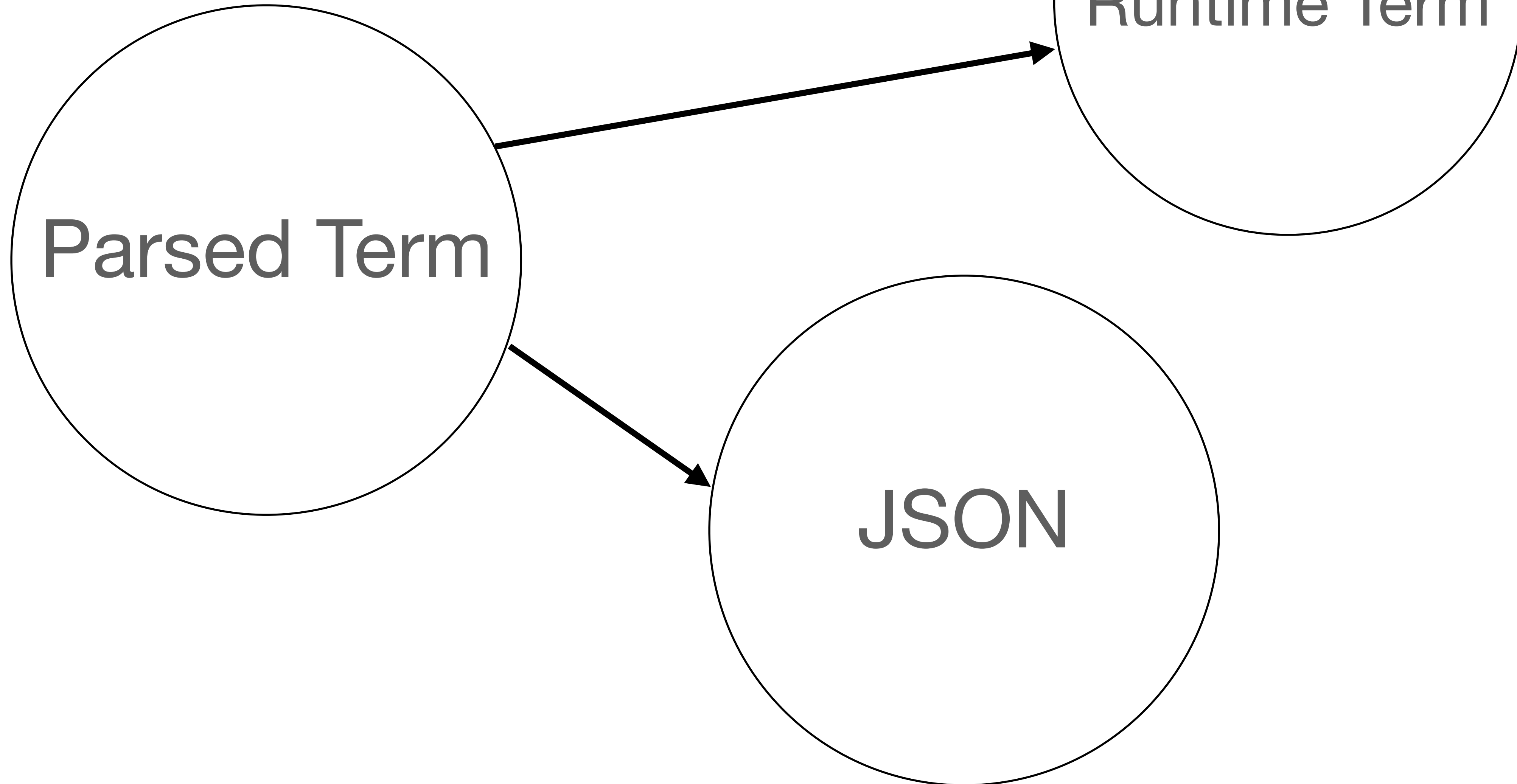
```
"#{mpcdec} #{process.quote(f)} -"
```

```
string.concat(
  [
    string(mpcdec),
    " ",
    string(process.quote(f)),
    "_ "
  ]
)
```

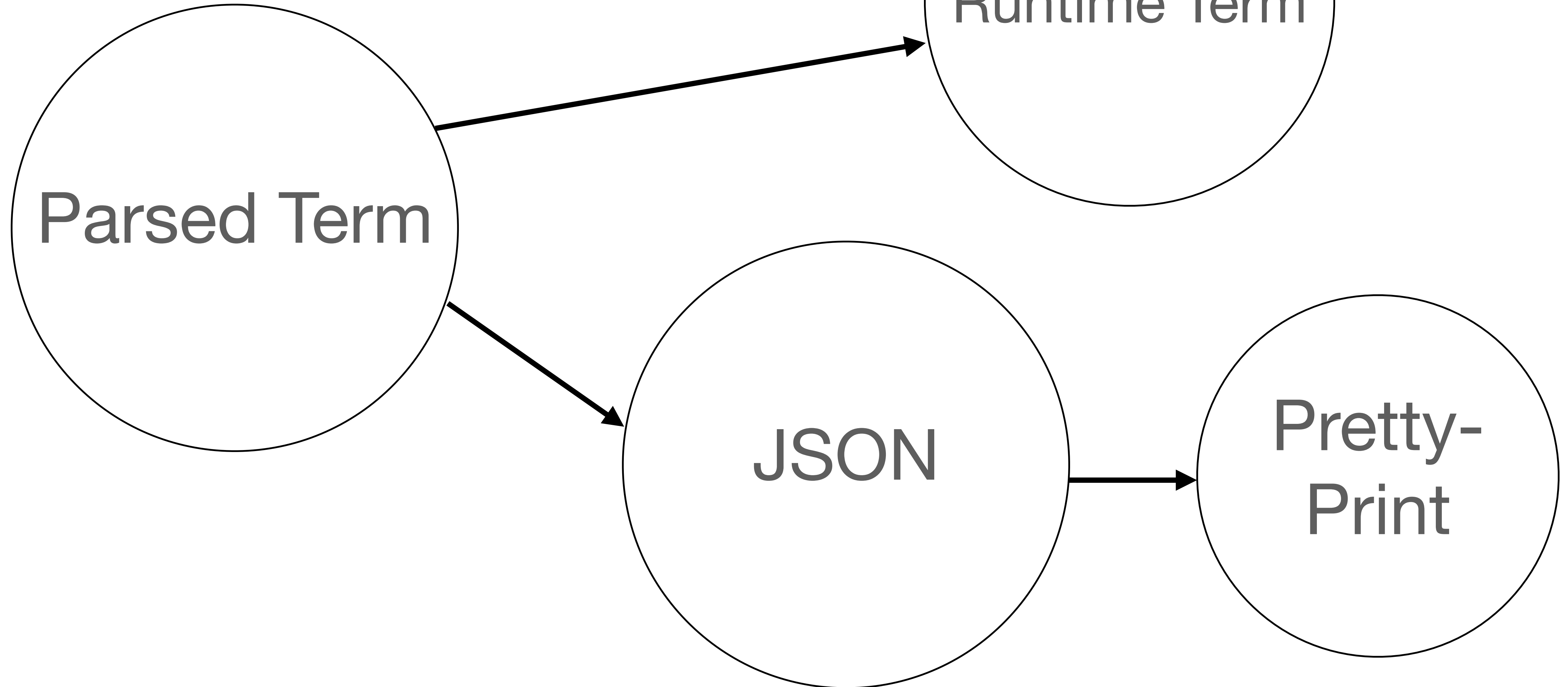

Prettier



Prettier




Prettier



Prettier

liquidsoap-prettier

1.5.3 • Public • Published 7 days ago

 [Readme](#)

 [Code](#) Beta

 [4 Dependencies](#)

 [0 Dependents](#)

 [10 Versions](#)

 [Settings](#)

Liquidsoap prettier

This package provides a **Prettier** plugin for liquidsoap code as well as a `liquidsoap-prettier` binary for formatting liquidsoap scripts.

Installation

`liquidsoap-prettier`

The `liquidsoap-prettier` command-line utility should be installed with this package and should be available following the usual node package binary conventions.

It works as follows:

```
$ liquidsoap-prettier [-w|--write] path/to/file.liq "path/with/glob/pattern"
```

Prettier plugin

The package also provides a prettier plugin which can be used to add liquidsoap script parsing to your project. To enable, you need a local `package.json`.

Install

```
> npm i liquidsoap-prettier
```

Weekly Downloads

97

Version

1.5.3

License

ISC

Unpacked Size

20 MB

Total Files

12

Last publish

7 days ago

Collaborators



Prettier

[formatter.nvim](#) / [lua](#) / [formatter](#) / [filetypes](#) / **liquidsoap.lua** 



toots Add formatter for liquidsoap script. ([#296](#))

Code

Blame

11 lines (9 loc) · 151 Bytes

```
1     local M = {}
2
3  ✓ function M.liquidsoap_prettier()
4     return {
5         exe = "liquidsoap-prettier",
6         args = {"--write"},
7         stdin = false
8     }
9     end
10
11    return M
```

Prettier

[formatter.nvim](#) / [lua](#) / [formatter](#) / [filetypes](#) / [liquidsoap.lua](#) 



toots Add formatter for liquidsoap script. ([#296](#))

Code

Blame

11 lines (9 loc) · 151 Bytes

```
1     local M = {}
2
3     function M.format()
4
5         exe = "liquidsoap-prettier",
6         args = {"--write"},
7         stdin = false
8     }
9     end
10
11    return M
```

Add support for your favorite editor!

Tree-sitter



[GitHub repository](#) 

[Introduction](#)

[Language Bindings](#)

[Parsers](#)


[Talks on Tree-sitter](#)

[Underlying Research](#)

[Using Parsers](#)




Introduction

Tree-sitter is a parser generator tool and an incremental parsing library. It can build a concrete syntax tree for a source file and efficiently update the syntax tree as the source file is edited. Tree-sitter aims to be:

- **General** enough to parse any programming language
- **Fast** enough to parse on every keystroke in a text editor
- **Robust** enough to provide useful results even in the presence of syntax errors
- **Dependency-free** so that the runtime library (which is written in pure [C](#) ) can be embedded in any application

Language Bindings

There are currently bindings that allow Tree-sitter to be used from the following languages:

- [C#](#) 
- [Go](#) 
- [Guide](#) 

Tree-sitter

```
_expr: $ =>  
  choice(  
    $.include,  
    $.if_def,  
    $.if_encoder,  
    $.if_version,  
    $.coerce,  
    $.parens,  
    $.integer,  
    $.not,  
    $.bool,  
    $.float,  
    $_.minus,  
    $.string,  
    $.var,
```


Tree-sitter

```
_expr: $ =>  
  choice(  
    $.include,  
    $.if_def,  
    $.if_encoder,  
    $.if_version,  
    $.coerce,  
    $.parens,  
    $.integer,  
    $.not,  
    $.bool,  
    $.float,  
    $_minus,  
    $.string,  
    $.var,
```

```
if_def: $ =>  
  seq(  
    choice("%ifdef", "%ifndef"),  
    choice($.var, $.subfield),  
    $_exprs,  
    optional(seq("%else", $_exprs)),  
    "%endif",  
  ),  
  
if_encoder: $ =>  
  seq(  
    choice("%ifencoder", "%ifnencoder"),  
    $.encoder,  
    $_exprs,  
    optional(seq("%else", $_exprs)),  
    "%endif",  
  ),
```

Tree-sitter

```
(source_file ; [0, 0] - [75, 0]
  (if_def ; [0, 0] - [14, 6]
    (var) ; [0, 7] - [0, 18]
    (comment) ; [1, 0] - [3, 18]
    (def ; [4, 0] - [13, 3]
      defined: (subfield ; [4, 4] - [4, 19]
        (var) ; [4, 4] - [4, 15]
        (method)) ; [4, 16] - [4, 19]
      arguments: (arglist ; [4, 19] - [4, 35]
        (labeled_argument ; [4, 20] - [4, 31]
          label: (var) ; [4, 21] - [4, 28]
          (string)) ; [4, 29] - [4, 31]
        (anonymous_argument ; [4, 33] - [4, 34]
          (var))) ; [4, 33] - [4, 34]
      (definition ; [5, 2] - [12, 5]
        (binding ; [5, 2] - [5, 21]
          defined: (var) ; [5, 2] - [5, 6]
          (definition ; [5, 8] - [5, 21]
            (app ; [5, 8] - [5, 21]
              name: (var)))) ; [5, 8] - [5, 19]
        (if ; [6, 2] - [12, 5]
          (if_condition ; [7, 4] - [7, 30]
            (not ; [7, 4] - [7, 30]
              (invoke ; [7, 8] - [7, 30]
                (invoke ; [7, 8] - [7, 18]
                  (var) ; [7, 8] - [7, 12]
                  (method)) ; [7, 13] - [7, 18]
```

Syntax tree for src/libs/utils.liq 4,1

:InspectTree

```
%ifndef environment
# Get the value of an environment variable.
# Returns `default` if the variable is not set.
# @category System
def environment.get(~default="", s) =
  env = environment()
  if
    not list.assoc.mem(s, env)
  then
    default
  else
    list.assoc(default=default, s, env)
  end
end
%endif

# Split the arguments of an url of the form `arg=bar&arg2
into
# `[("arg","bar"),("arg2","bar2")]`. The returned strings
coded (see
# `url.decode`).
# @category String
# @param args Argument string to split.
def url.split_args(args) =
  def f(x) =
    ret = r/=/.split(x)
    arg = url.decode(list.nth(default="", ret, 0))
```

Top src/libs/utils.liq

2,1

Tree-sitter

```
(source_file ; [0, 0] - [3, 0]
  (var) ; [0, 0] - [0, 1]
  (ERROR) ; [0, 1] - [0, 2]
  (infix ; [0, 2] - [0, 11]
    (string) ; [0, 2] - [0, 7]
    (op) ; [0, 8] - [0, 9]
    (integer)) ; [0, 10] - [0, 11]
  (app ; [2, 0] - [2, 12]
    name: (var) ; [2, 0] - [2, 5]
    (anonymous_arg ; [2, 6] - [2, 11]
      (string)))) ; [2, 6] - [2, 11]
```

```
x."foo" + 1
print("bla")
```

Live Demo!

Lezer

- Rince, repeat
- Used by code-mirror
- Used by AzuraCast!

Lezer

```
expr {  
  Include |  
  IfDef |  
  IfEncoder |  
  IfVersion |  
  Coerce |  
  Parens |  
  Integer |  
  Not |  
  Bool |
```

Lezer

```
expr {  
  Include |  
  IfDef |  
  IfEncoder |  
  IfVersion |  
  Coerce |  
  Parens |  
  Integer |  
  Not |  
  Bool |
```

```
IfDef {  
  (kw<"%ifdef"> | kw<"%ifndef">)  
  (Var | Subfield)  
  exprs  
  (kw<"%else"> exprs)? kw<"%endif">  
}
```


Lezer

Liquidsoap playground

```
1 # ✨ Welcome to liquidsoap's online interpreter! ✨
2 # 🤖 Language version: 2.3.0+git@958c6ef30
3 # Write your code here:
4
5 def url.split_args(args) =
6   def f(x) =
7     ret = r/=/.split(x)
8     arg = url.decode(list.nth(default="", ret, 0))
9     val = url.decode(list.nth(default="", ret, 1))
10    (arg, val)
11   end
12
13   l = r/&/.split(args)
14   list.map(f, l)
15 end
```


VSCode

- Rince, repeat
- Used by a lot of programmers
- Regexp-based grammar

VSCode

```
136     },
137     "expressions": {
138         "patterns": [
139             {
140                 "comment": "inline if",
141                 "begin": "(\\?)(?![\\?\\.])",
142                 "end": "(:)",
143                 "beginCaptures": { "1": { "name": "keyword.control.liquidsoap" } },
144                 "endCaptures": { "1": { "name": "keyword.control.liquidsoap" } },
145                 "patterns": [{ "include": "#expressions" }]
146             },
147             {
148                 "comment": "if",
149                 "begin": "\\b(if)\\b",
150                 "end": "\\b(end)\\b",
151                 "beginCaptures": { "1": { "name": "keyword.control.liquidsoap" } },
152                 "endCaptures": { "1": { "name": "keyword.control.liquidsoap" } },
153                 "patterns": [
154                     {
155                         "match": "\\b(then|else|elsif)\\b",
156                         "name": "keyword.control.liquidsoap"
157                     },
158                     { "include": "$self" }
159                 ]
160             },

```

VSCode

The image shows a screenshot of the Visual Studio Code (VS Code) editor interface. On the left, the Explorer sidebar shows a project named 'AUTOCUE' with several files, including 'test_autocue.cue_file.liq' which is selected. The main editor window displays the content of this file, which is a cue file script. The script includes comments and code for handling metadata and playback gain. The code is as follows:

```
90 #radio = amplify(1.,override=replaygain_track_gain,radio)
91 # AzuraCast already has the following line enabled:
92 #radio = amplify(1.,override="liq_amplify",radio)
93
94 # Show metadata in log
95 def show_meta(m)
96     label="show_meta"
97     l = list.sort.natural(metadata.cover.remove(m))
98     list.iter(fun(v) -> log(level=4, label=label, "#{v}"), l)
99     nowplaying = ref(m["artist"] ^ " - " ^ m["title"])
100     if m["artist"] == "" then
101         if string.contains(substring=" - ", m["title"]) then
102             let (a, t) = string.split.first(separator=" - ", m["title"])
103             nowplaying := a ^ " - " ^ t
104         end
105     end
106
107 # show `liq_` & other metadata in level 3
108 def fl(k, _) =
109     tags = ["duration", "kind", "replaygain_track_gain", "replaygain_referenc
110     string.contains(prefix="liq_", k) or list.mem(k, tags)
111 end
112 liq = list.assoc.filter((fl), l)
113 list.iter(fun(v) -> log(level=3, label=label, "#{v}"), liq)
114
115 log(level=3, label=label, "Now playing: #{nowplaying()}")
116 if m["liq_amplify"] == "" then
117     log(level=2, label=label, "Warning: No liq_amplify found, expect loudness
118 end
119 if m["liq_blank_skipped"] == "true" then
120     log(level=2, label=label, "Blank (silence) detected in track, ending earl
121 end
122 end
```

Github

- Rince, repeat
- Syntax Highlighting
- Regexp-based grammar



Add language: liquidsoap #6565

Open toots wants to merge 1 commit into `github-linguist:master` from `toots:master`

Conversation 4 Commits 1 Checks 0 Files changed 10



toots commented on Oct 5, 2023

Description

This PR adds support for the liquidsoap language. The language has been existing since ~2005 and is widely used to run media streaming applications. Although its original scope is specialized, the language itself is a general-purpose scripting language that is functional and statically typed with inferred types.

Checklist:

- I am adding a new language.
 - The extension of the new language is used in hundreds of repositories on GitHub.com.
 - Search results for each extension:
 - https://github.com/search?type=code&q=NOT+is%3Afork+path%3A*.liq
 - I have included a real-world usage sample for all extensions added in this PR:
 - Sample source(s):
 - liquidsoap standard library: <https://github.com/savonet/liquidsoap/tree/main/src/libs>

Github



lildude requested changes on Oct 5, 2023

[View reviewed changes](#)

lildude left a comment

Member



Given its age, usage is surprisingly low in GitHub at the moment, but that could be due to old inactive repos that haven't been indexed yet (the new search will get to them one day).

Other than the suggestion, your three samples suppressed in the diff are too big (probably because of the extensive comments). Please replace these with smaller yet diverse samples with fewer comments.

As popularity isn't sufficient for inclusion right now, I'll label it as such and will review popularity with each release.



GitHub



lildude requested changes on Oct 5, 2023

[View reviewed changes](#)

lildude left a comment

Member ...

Given its age, usage is surprisingly low in GitHub at the moment. It has not been indexed yet (the new search engine is still in beta).

Other than that, the changes in the diff are too big (probably because of the extensive comments). Please replace these with smaller yet diverse samples with fewer comments.

As popularity isn't sufficient for inclusion right now, I'll label it as such and will review popularity with each release.



Please commit your script files to GitHub!

More..?

- Discord: highlight.js
- Linting?
- What else?
- Community help!

Liquidsoap 2.3.x internal changes

- Streaming loop
- Clocks

Streaming Loop

- Replaced a pre-allocated shared frame model with a extensible content frame
- Huge cleanup in application logic
- Better integration with external tools (e.g. FFmpeg)

Streaming Loop

`single("/path/to/file.mp3")`



`playlist("/path/to/directory/")`

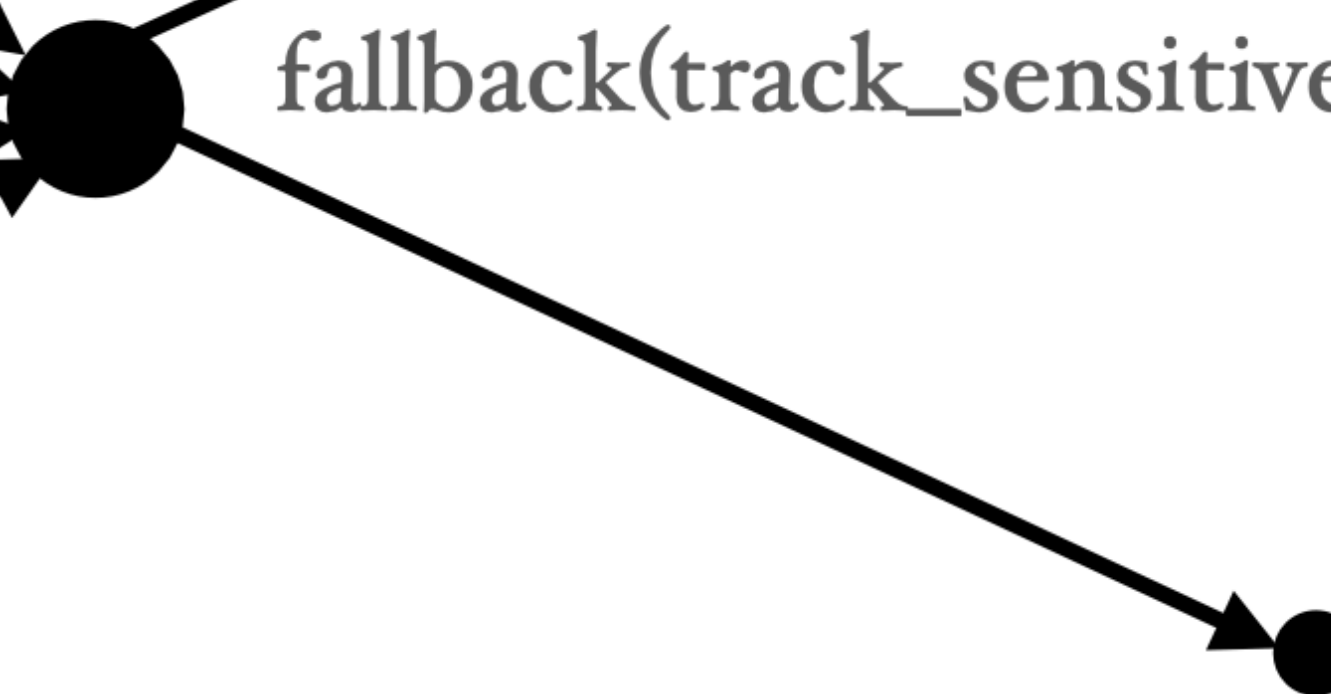
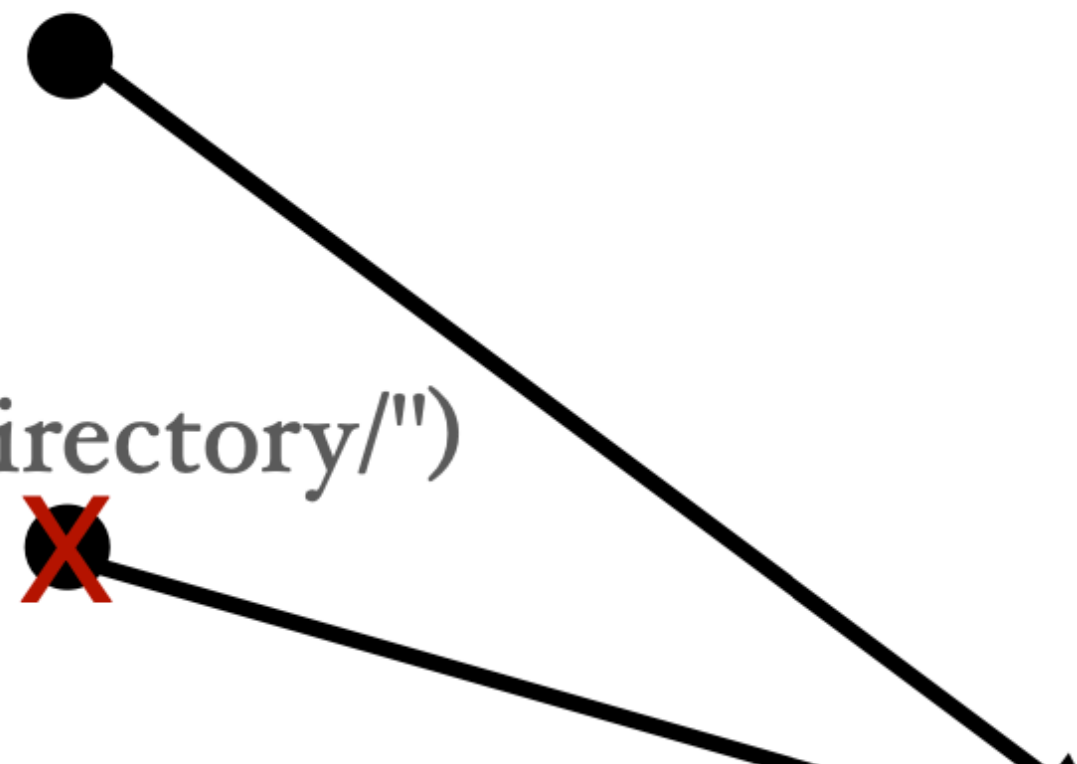
`request.queue(...)`

`input.harbor(...)`

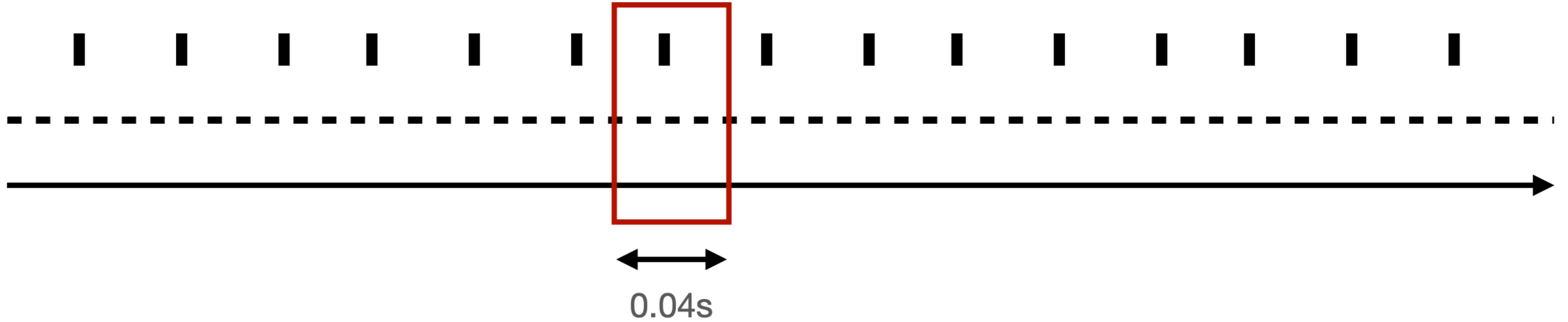
`fallback(track_sensitive=false, ...)`

`output.file(...)`

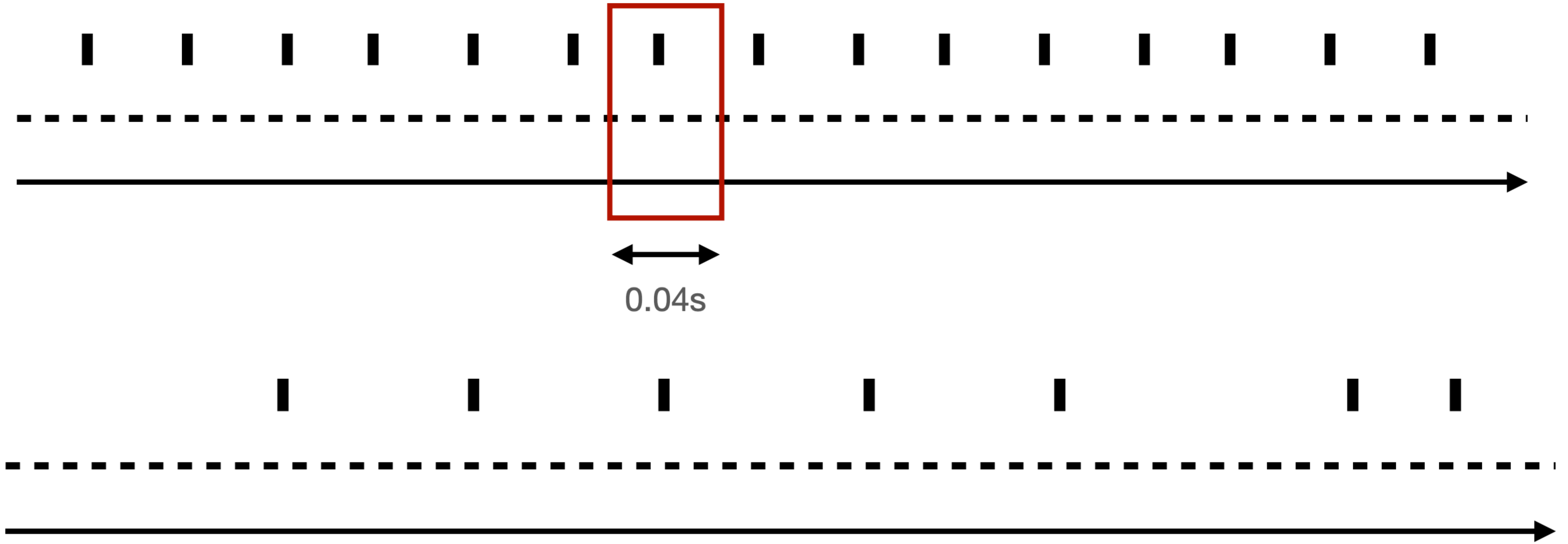
`output.icecast(...)`



Streaming Loop



Streaming Loop

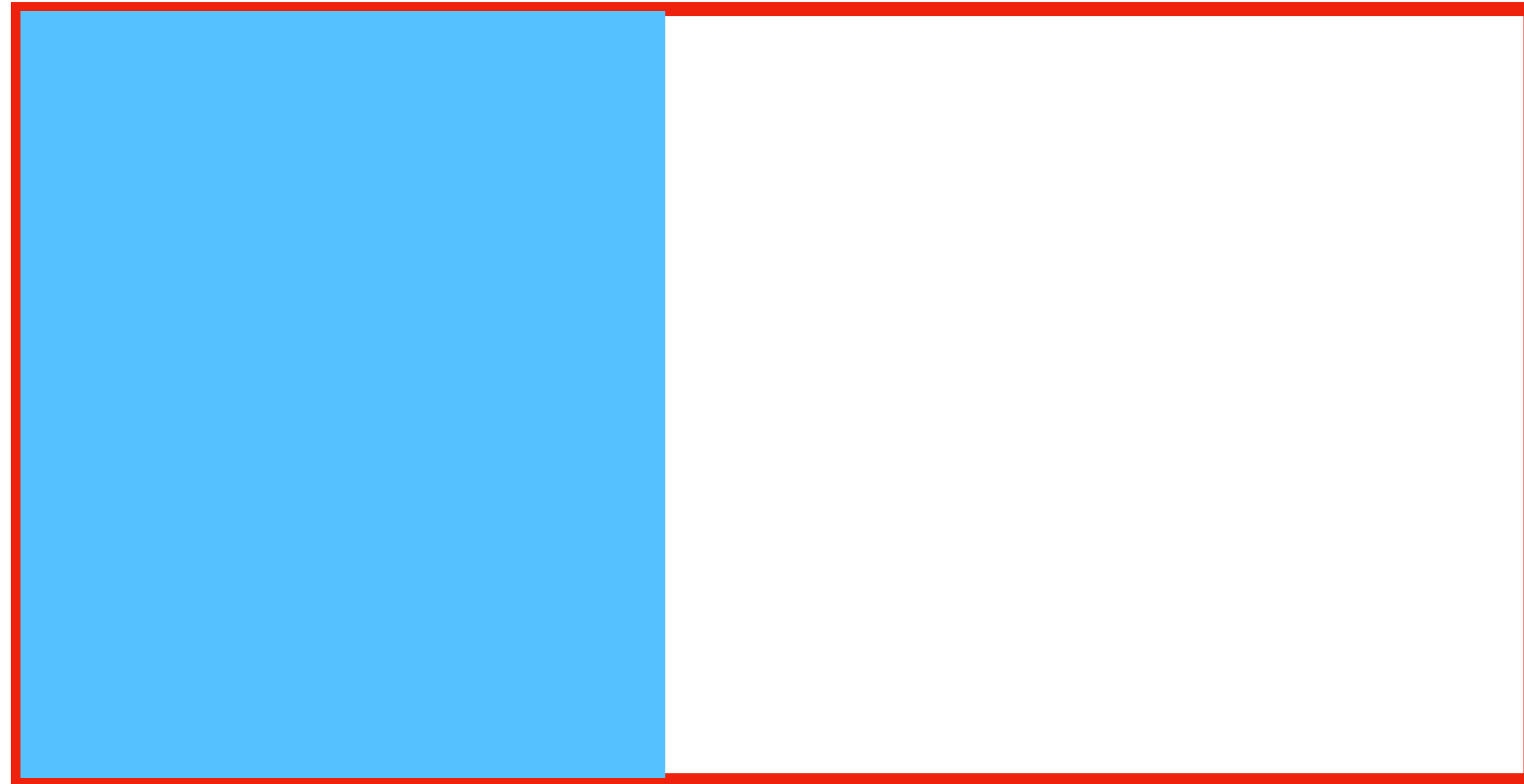


Streaming Loop



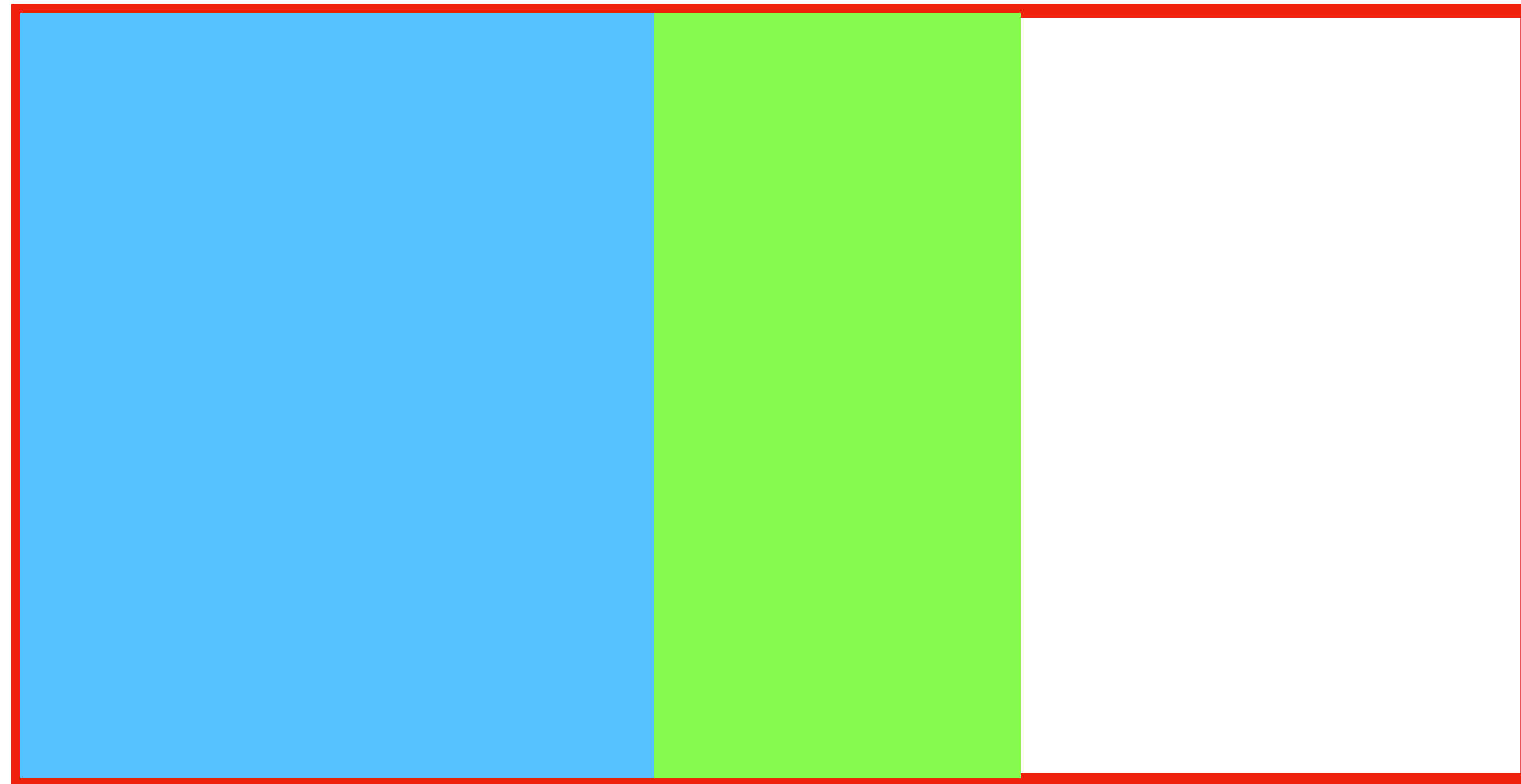
0.04s

Streaming Loop



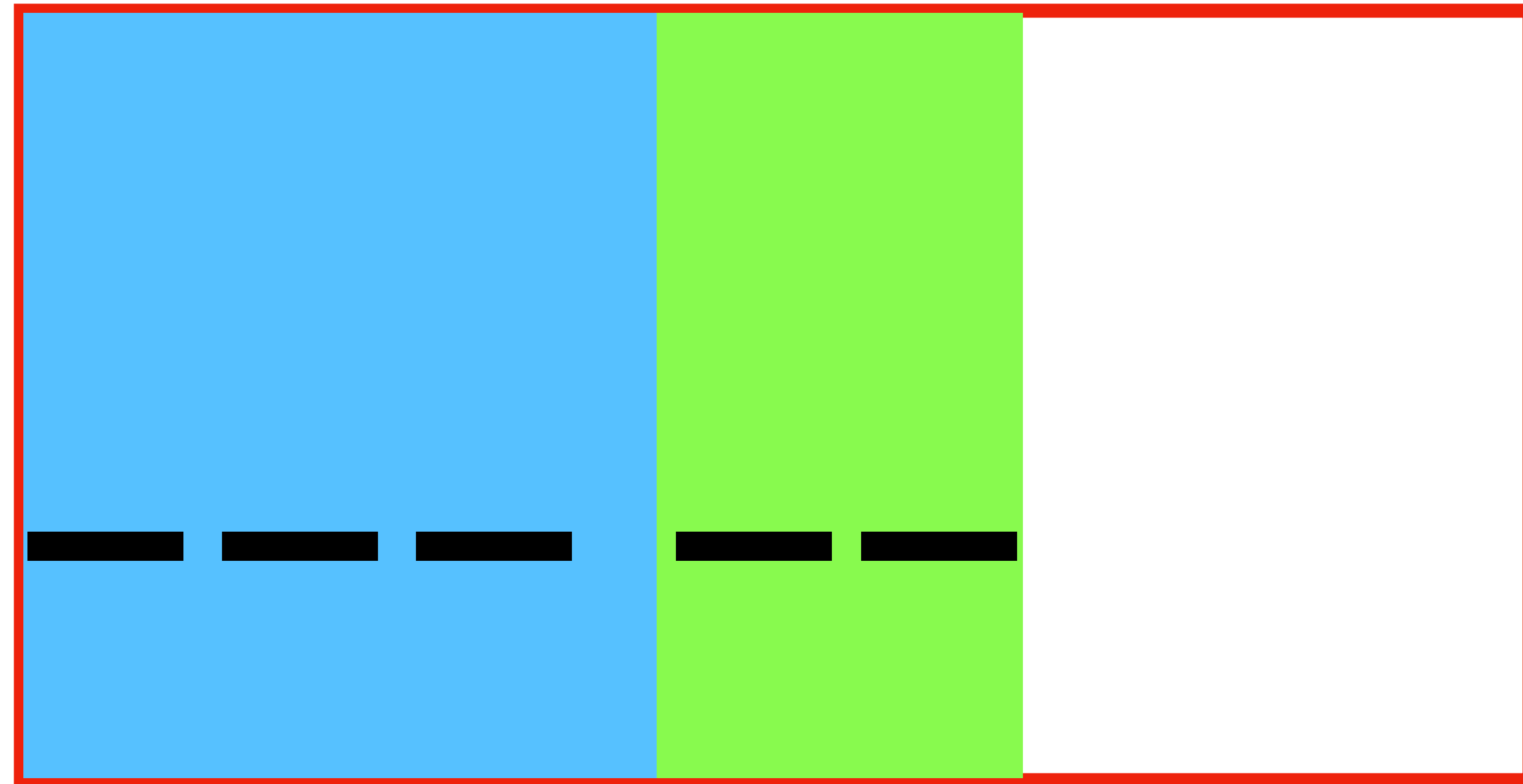
0.04s

Streaming Loop



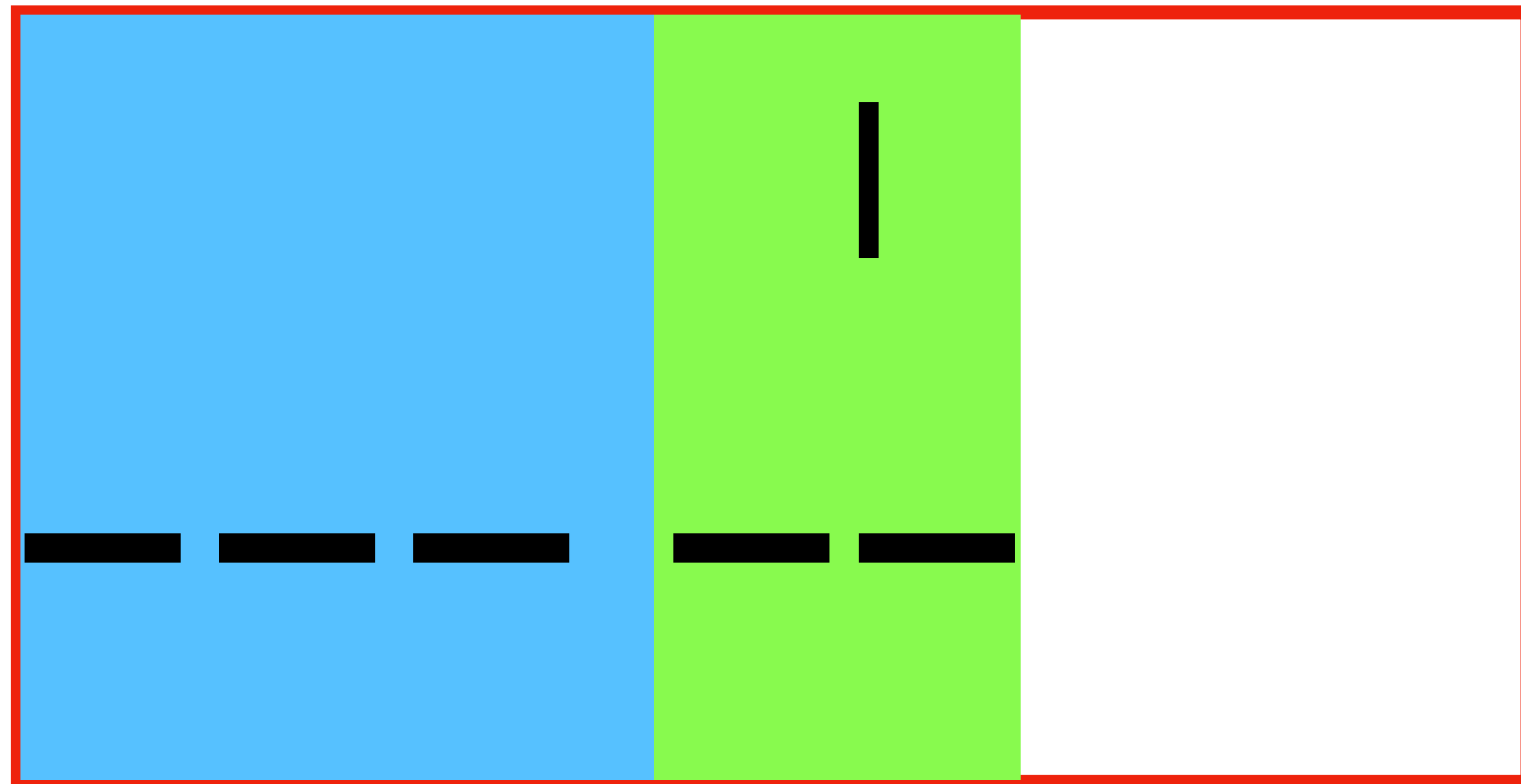
0.04s

Streaming Loop



0.04s

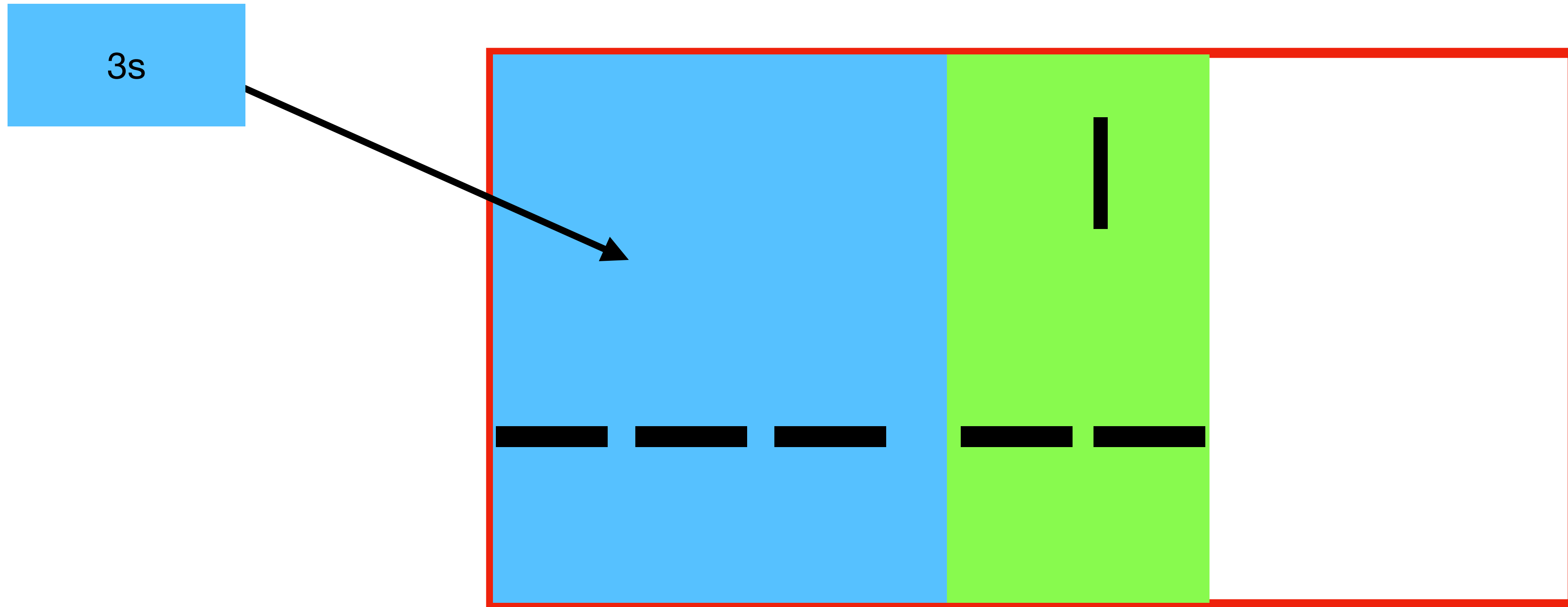
Streaming Loop



0.04s

Streaming Loop

Decoded chunk

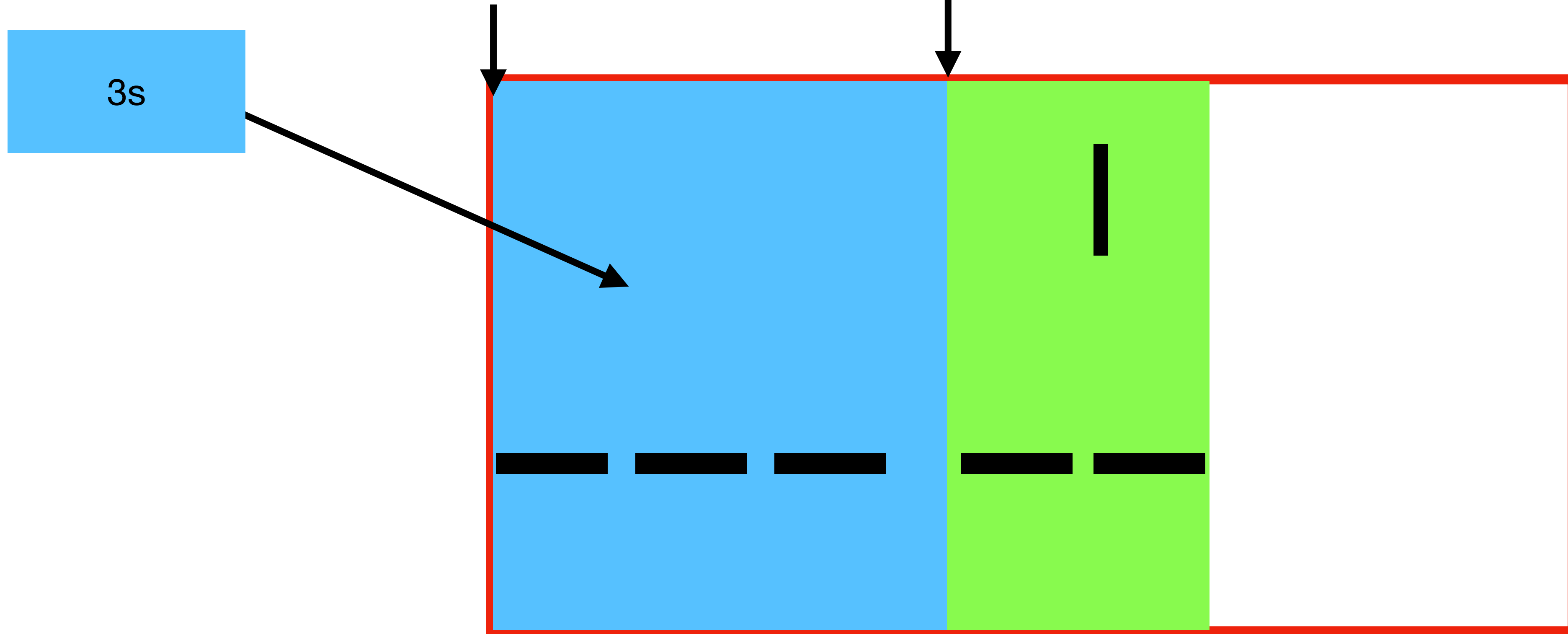


0.04s

Streaming Loop

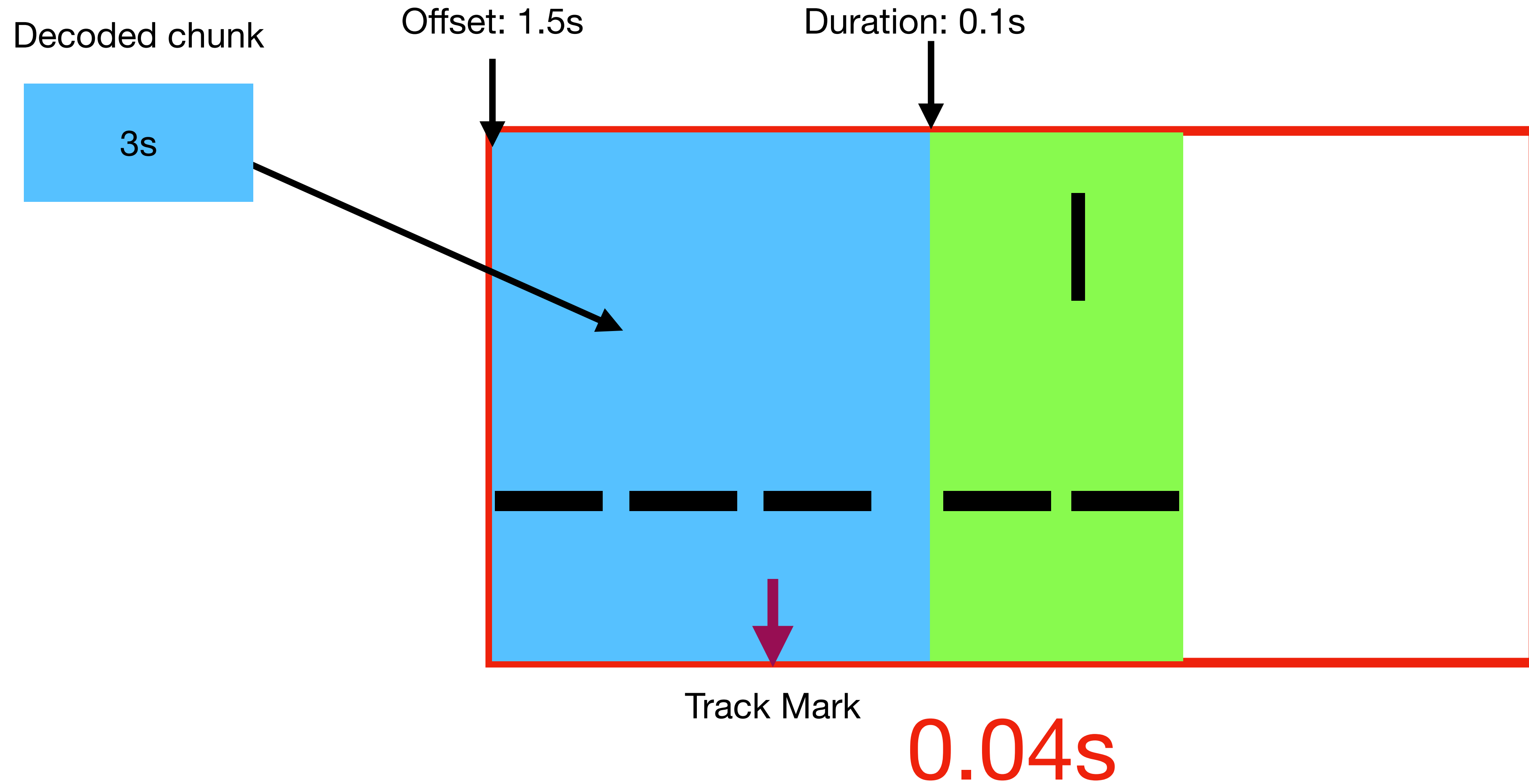
Decoded chunk

3s

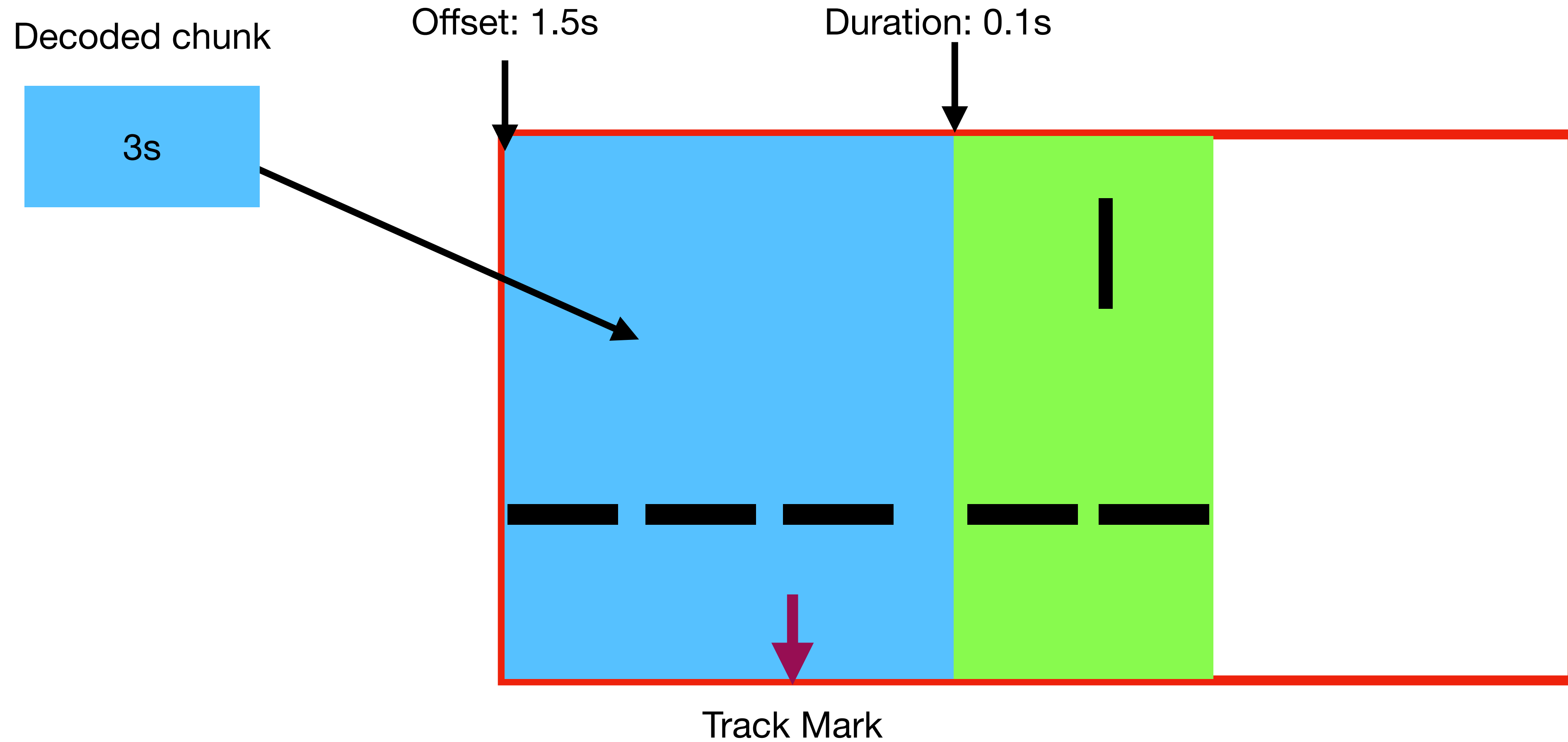


0.04s

Streaming Loop



Streaming Loop



No more bugs with breaks vs. track marks!

Clocks

- Reworked the internal logic to share code
- Cleanup original implementation assumptions that turned out unused
- Rationalize and expose clocks
- Self-sync? Clock safe?
- Expose better tools w.r.t. clocks
- Future: let user run their own clock in the script?

Clocks

```
s = input.http("http://wwoz-sc.streamguys1.com/wwoz-hi.mp3")  
output.ao(fallible=true, s)
```


Clocks

```
2024/05/20 14:23:04 >>> LOG END
```

```
Fatal error: exception At ../../src/libs/extra/telnet.liq, line 204, char 6-40:
```

```
s = input.http(%argsof(input.http), s)
```

```
Error 17: clock generic has multiple synchronization sources. Do you need to set self_sync=false?
```

```
Sync sources:
```

```
source(id=input.http) from source input.http
```

```
ao from source ao
```

```
Stack traces:
```

```
clock generic:
```

```
at ../../src/libs/extra/telnet.liq, line 204, char 6-40
```

```
at /tmp/ao.liq, line 1, char 4-60
```

```
input.http:
```

```
at ../../src/libs/extra/telnet.liq, line 204, char 6-40
```

```
at /tmp/ao.liq, line 1, char 4-60
```

```
ao:
```

```
at /tmp/ao.liq, line 3, char 0-27
```

Live Demo!

OCaml 5 and multi-core support

OCaml 5 and multi-core support

ocaml / ocaml

[Code](#) [Issues 286](#) [Pull requests 281](#) [Discussions](#) [Actions](#) [Projects](#) [Security](#) [Insights](#)

Regression with default GC settings between 4.14.2 and 5.1.1 #13123

Open toots opened this issue on Apr 25 · 7 comments



toots commented on Apr 25

Contributor

Hi!

We're in the process of switching `liquidsoap` to OCaml `5.1.1` and we've noticed some pretty severe regressions with the garbage collector.

I'm still testing and need to confirm whether or not we can get to comparable memory/CPU usage than with `4.14` but I'd like to report a first, very simple case.

This liquidsoap script:

```
output.dummy(blank())
```

Is basically a runtime loop creating a `float array array` (ocaml native) of `0.04s` of blank PCM audio every `0.04` and discarding it. No C code, only OCaml.

With `4.14`, the memory footprint looks like this:

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

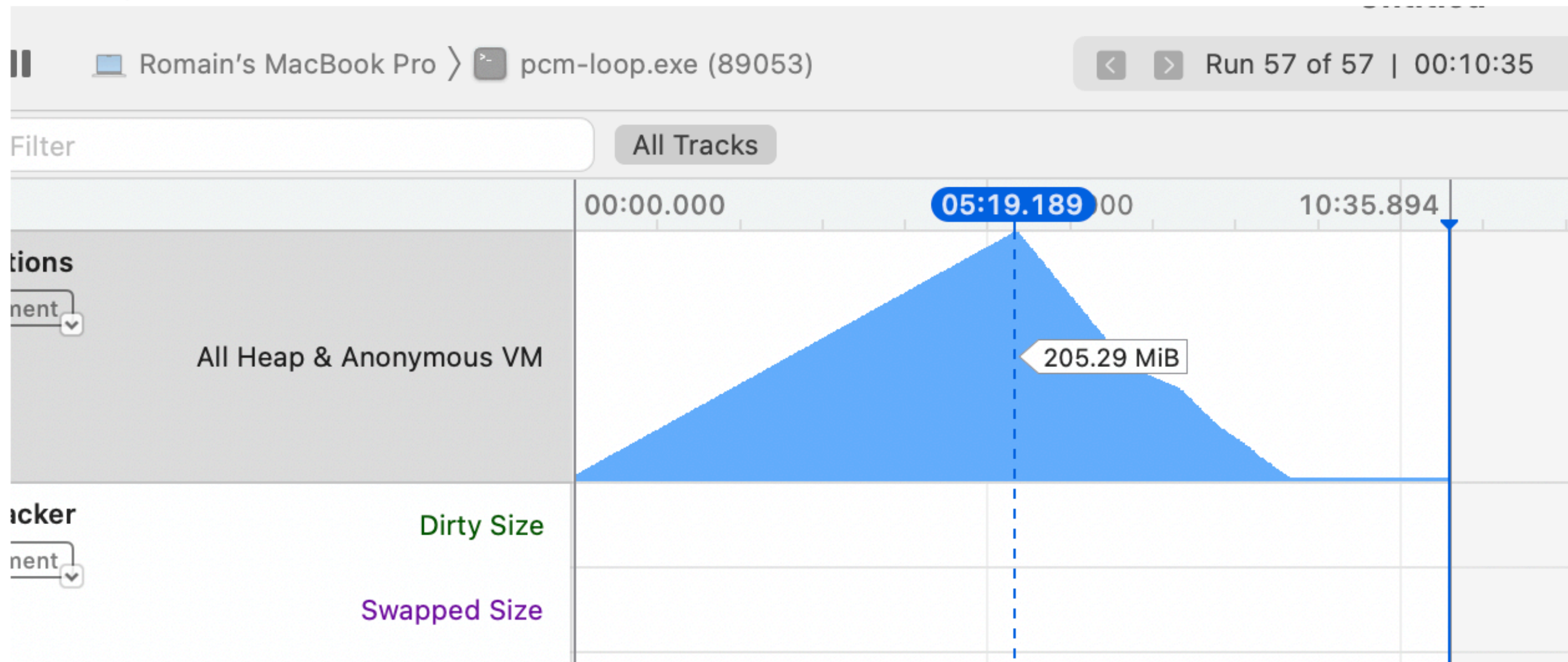
No milestone

Development

No branches or pull re

OCaml 5 and multi-core support

Memory consumption:



Woof!

Looks like I can see. the following:

- The GC is kinda doing what it needs to do in my previous example: keep a ratio of the live memory, here, about 65Mo out of the ~40Mo of allocated memory. It's the spirit of it but this is neither what 4.14 was effectively doing and, frankly, not a great experience when you work with an app that has large amount of permanently allocated memory and short amount of transitory memory.

OCaml 5 and multi-core support



toots commented on [1a262d4](#) 3 weeks ago

Member

Author



Thanks!



vitoyucepi commented on [1a262d4](#) 3 weeks ago • edited

Collaborator



And here's a comparison between 5.1.1 and 4.14.2.

The script consists only of metrics and a sine stream to icecast using libopus.

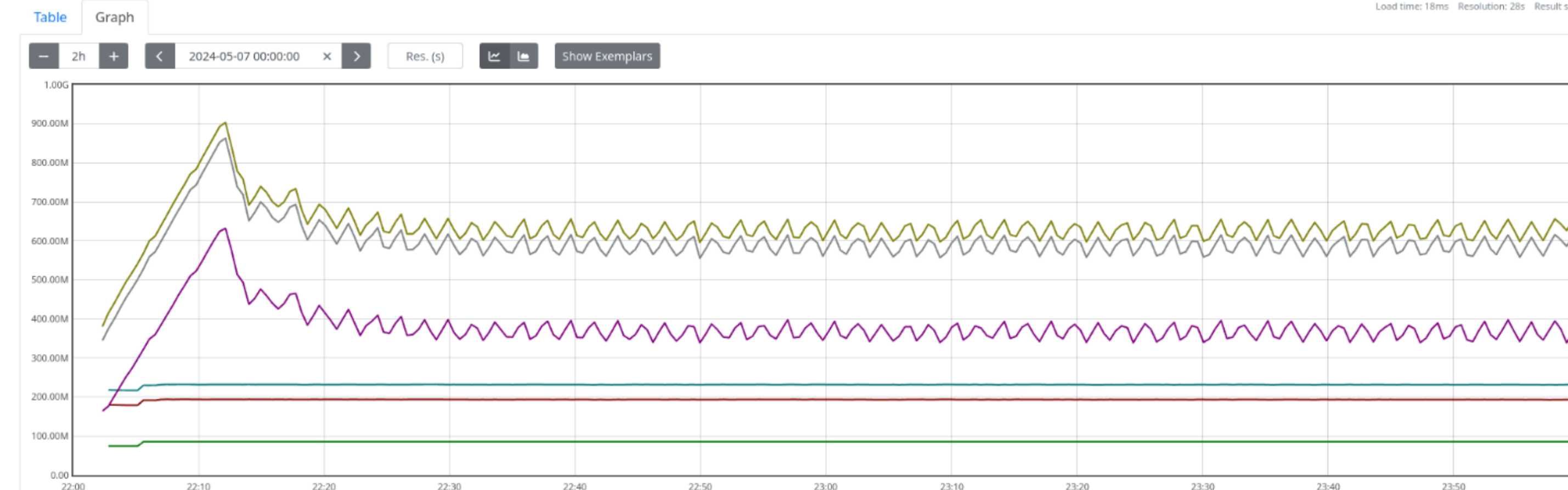
[ac425d0](#)

Use local time Enable query history Enable autocomplete Enable highlighting Enable linter

liquidsoap_process_memory_bytes{type=~"physical|private|managed"}

Execute

Load time: 18ms Resolution: 28s Result series: 6



OCaml 5 and multi-core support

- New memory management model not mature enough
- Unsure exactly how to leverage multi-core support
- Multi-processes vs. single script
- Seems more opportunistic to optimize existing memory and CPU usage!

Optimizations

Optimizations

- High memory usage from the standard library
- High CPU usage due to language complexity at runtime

Reduce CPU usage at runtime

Reduce CPU usage at runtime

Parse Script

```
list.add(x, 1)
```

Reduce CPU usage at runtime

Parse Script

```
list.add(x, 1)
```

```
`App  
  ( `Invoke {  
    invoked = `Var "list";  
    meth = "add"  
  },  
  [("", `Var "x"); ("", `Var "1")] )
```

Reduce CPU usage at runtime

Parse Script

Type-check

Variable "list" exists?

```
`App  
  ( `Invoke {  
    invoked = `Var "list";  
    meth = "add"  
  },  
  [("", `Var "x"); ("", `Var "l")] )
```

Reduce CPU usage at runtime

Parse Script

Type-check

Variable "list" exists?

```
`App  
  ( `Invoke {  
    invoked = `Var "list";  
    meth = "add"  
  },  
  [("", `Var "x"); ("", `Var "l")] )
```

Method "add" present?

Reduce CPU usage at runtime

Parse Script

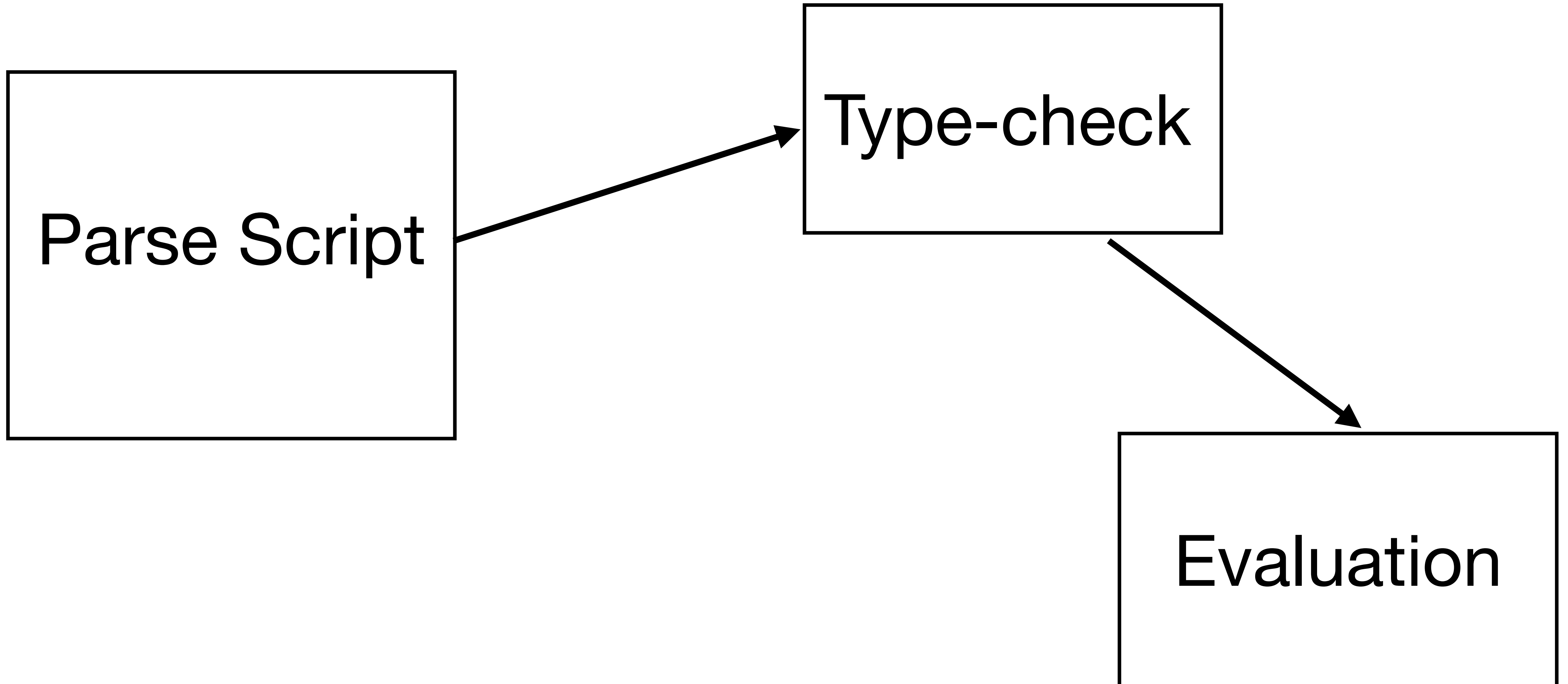
Type-check

```
`App  
  ( `Invoke {  
    invoked = `Var "list";  
    meth = "add"  
  },  
  [( "", `Var "x"); ( "", `Var "l" ) ] )
```

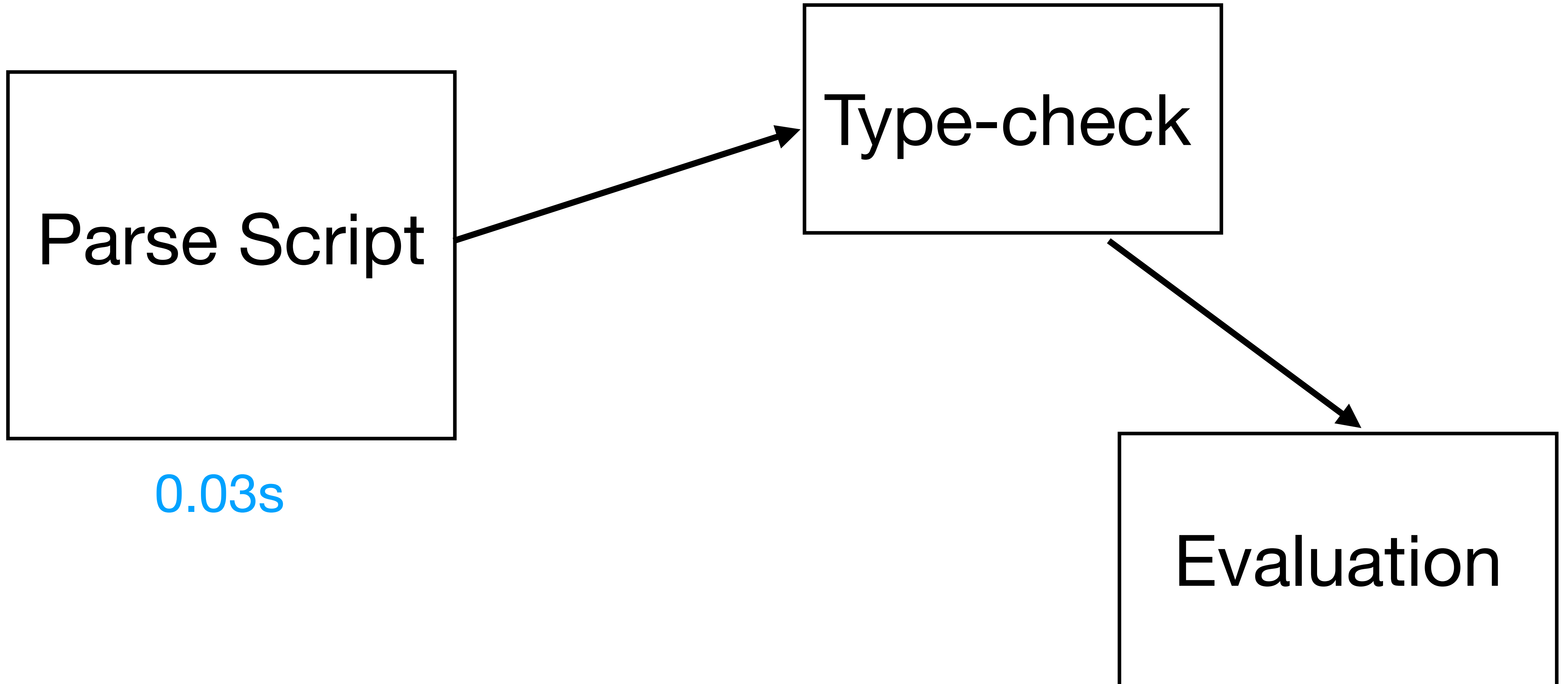
Variable "list" exists?

Method "add" present?

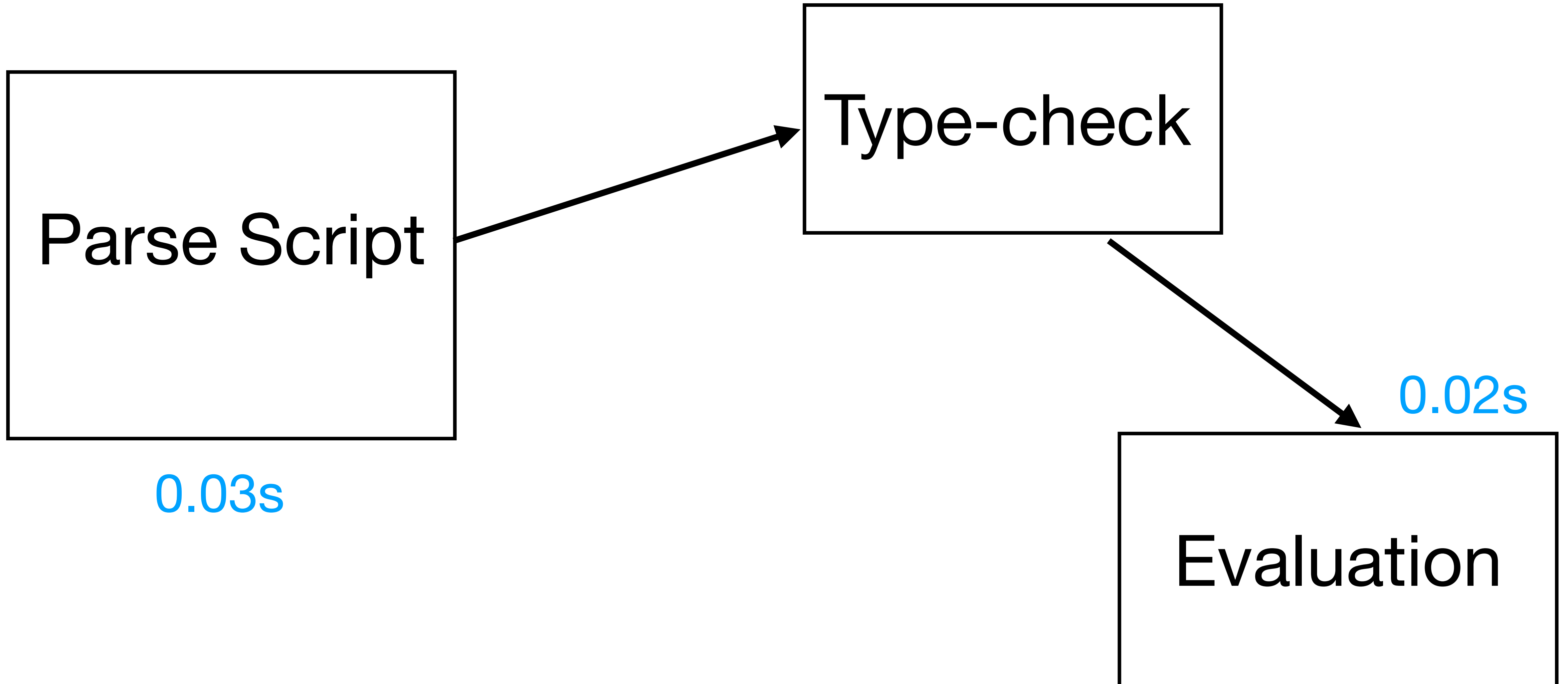
Reduce CPU usage at runtime



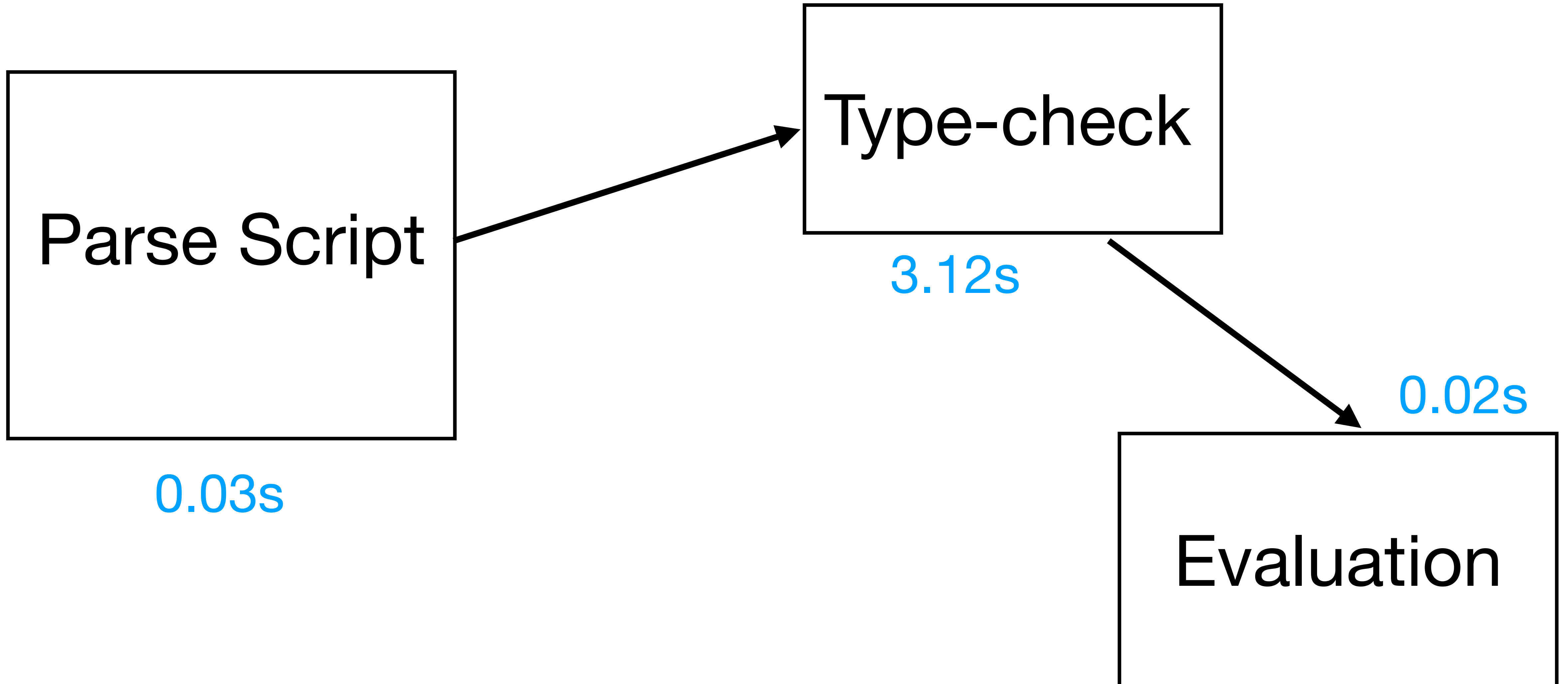
Reduce CPU usage at runtime



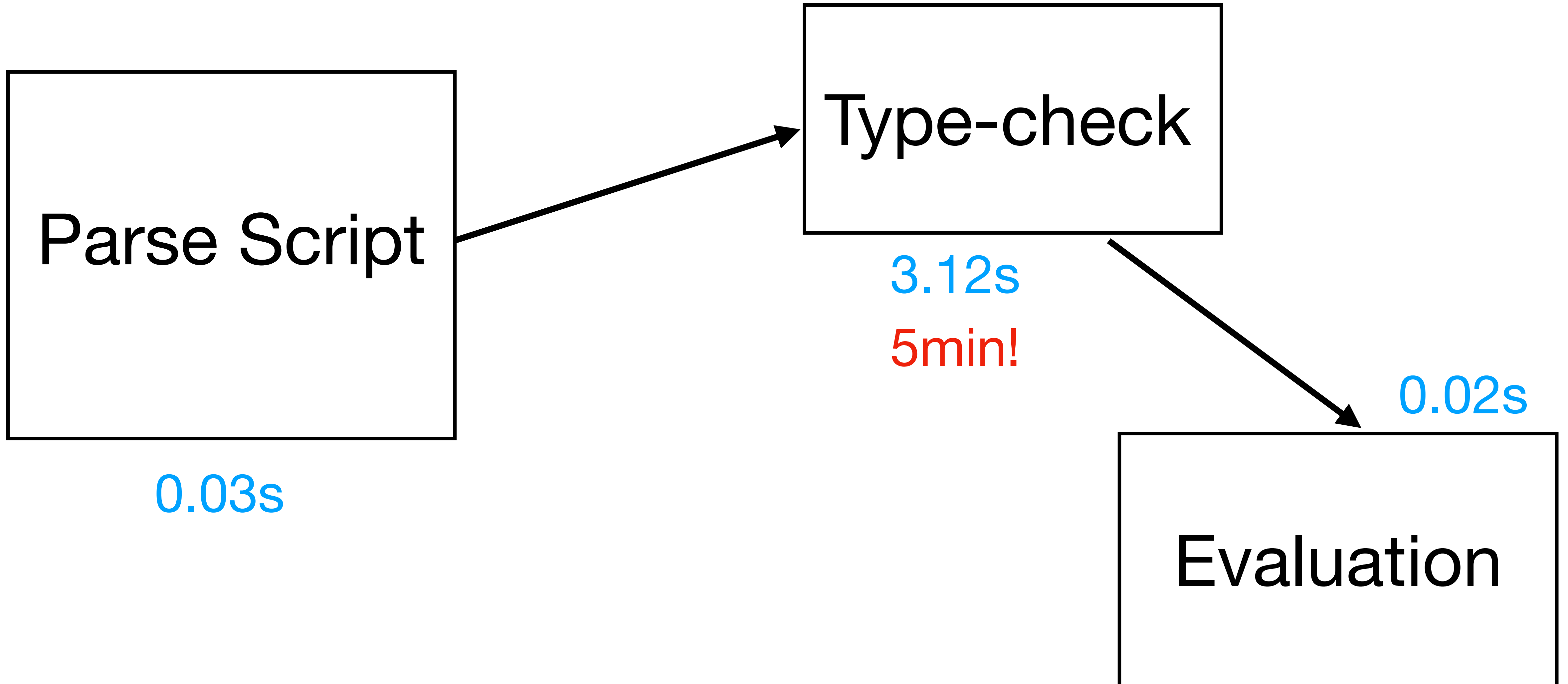
Reduce CPU usage at runtime



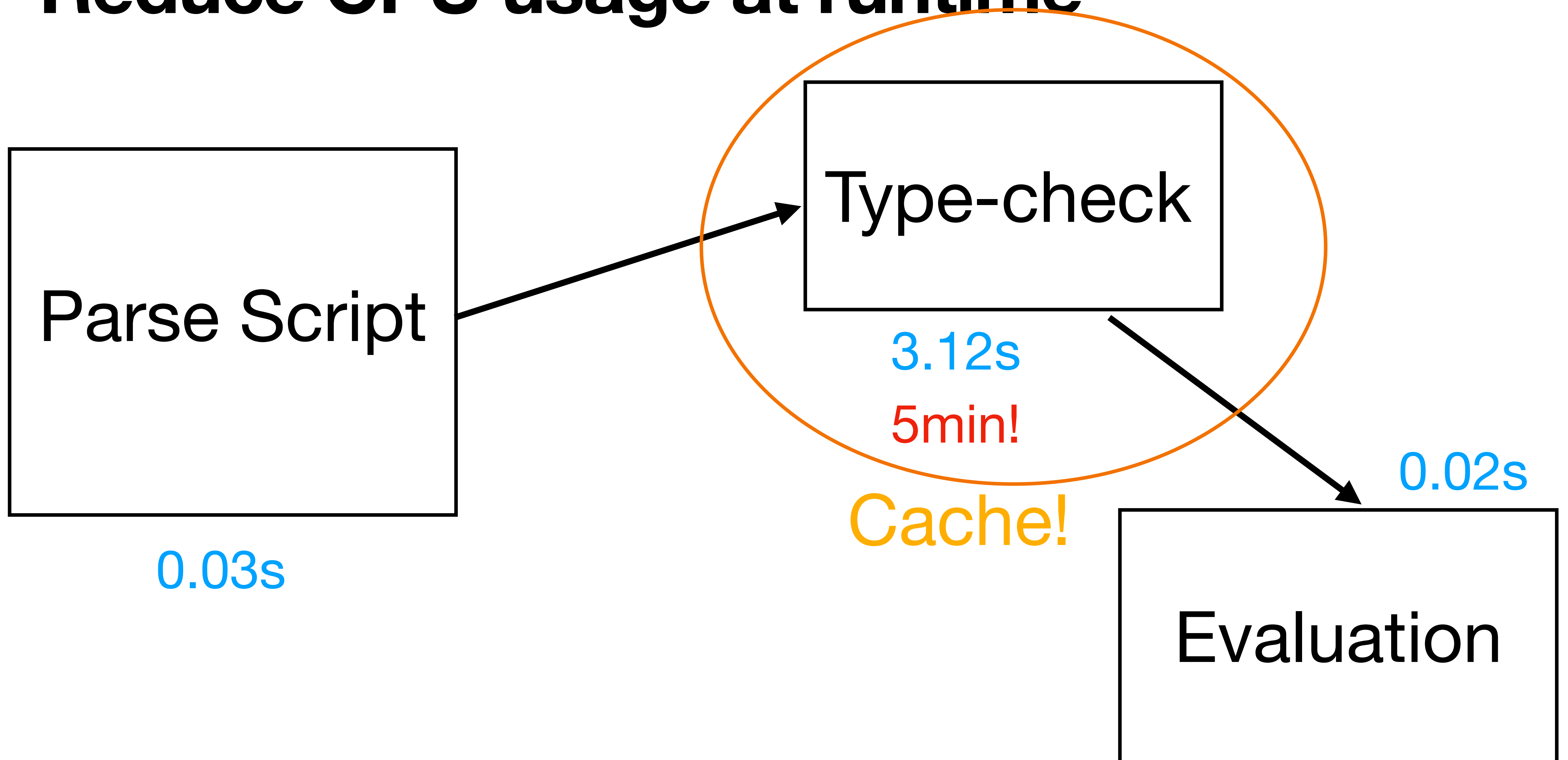
Reduce CPU usage at runtime



Reduce CPU usage at runtime



Reduce CPU usage at runtime



Reduce CPU usage at runtime

Cache terms #3924

 Open toots wants to merge 2 commits into `main` from `cache-term` 

 Conversation 0

 Commits 2

 Checks 26

 Files changed 58



toots commented last week · edited ▾

Member ...

This PR introduces a cache for running scripts!

Functionality

During the first execution, the script is parsed, type checked and evaluated. On second and any following execution, a cache of the script is used, reducing the typechecking phase by a 100x factor!

First execution (m3 macbook):

```
39 [startup:3] Ffmpeg bitstream filters regis
39 [startup:3] Liquidsoap binary changed: cd
39 [startup:3] Cache retrieval: 0.03s
39 [startup:3] Typechecking: 2.56s
39 [startup:3] Evaluation: 0.01s
39 [clock:3] Starting clock generic with 2 s
```

Second execution:

```
43 [startup:3] Ffmpeg bitstream filters regis
43 [startup:3] Loading script from cache!
43 [startup:3] Cache retrieval: 0.03s
43 [startup:3] Evaluation: 0.01s
43 [clock:3] Starting clock generic with 2 sou
```

This will greatly improve runtime performances for most users. We have had reports of startup time taking up to 5 minutes

Reduce memory usage

- Compiling phase
- Propagation of variables

Reduce memory usage

- Compiling phase
- Propagation of variables

```
list.add(x, 1)
```


Reduce memory usage

- Compiling phase
- Propagation of variables

```
# list;;
- :
  {
    add : 'a.('a, ['a]) -> ['a],
    append : 'a.(['a], ['a]) -> ['a],
    assoc : 'a.'b.((?default : 'a?, 'b, ['b * 'a]) -> 'a)
    .{
      filter : 'c.'d.(((c, 'd) -> bool), ['c * 'd]) -> ['c * 'd],
      filter_map : 'c.'d.'e.(((c, 'd) -> 'e?), ['c * 'd]) -> ['e],
      mem : 'c.'d.(c, ['c * 'd]) -> bool,
      nullable : 'c.'d.(c, ['c * 'd?]) -> 'd?,
      remove : 'c.'d.((c, ['c * 'd]) -> ['c * 'd])
      .{all : 'e.'f.(e, ['e * 'f]) -> ['e * 'f]
    }
  },
  case : 'a.'b.(['a], 'b, ((c, ['a]) -> 'b)) -> 'b,
  cons : 'a.('a, ['a]) -> ['a],
  dcase : 'a.'b.(['a], (c -> 'b), ((c, ['a]) -> 'b)) -> 'b,
  exists : 'a.(((c) -> bool), ['a]) -> bool,
  filter : 'a.(?remove : ((c) -> unit), ((c) -> bool), ['a]) -> ['a],
  filter_map : 'a.'b.(((c) -> 'b?), ['a]) -> ['b],
  find : 'a.(?default : 'a?, ((c) -> bool), ['a]) -> 'a,
  flatten : 'a.([[a]]) -> ['a],
  fold : 'a.'b.(((c, 'b) -> 'a), 'a, ['b]) -> 'a
  .{right : 'c.'d.(((c, 'd) -> 'd), 'd, ['c]) -> 'd
},
  for_all : 'a.(((c) -> bool), ['a]) -> bool,
  hd : 'a.(?default : 'a?, ['a]) -> 'a,
  ind : 'a.'b.(['a], 'b, ((c, ['a], 'b) -> 'b)) -> 'b,
  index : 'a.(((c) -> bool), ['a]) -> int,
  indexed : 'a.(['a]) -> [int * 'a],
  init : 'a.(int, ((int) -> 'a)) -> ['a],
  insert : 'a.(int, 'a, ['a]) -> ['a],
  is_empty : 'a.(['a]) -> bool,
  iter : 'a.(((c) -> unit), ['a]) -> unit,
  iterator : 'a.(['a?]) -> () -> 'a?,
  iteri : 'a.(((int, 'a) -> unit), ['a]) -> unit,
  last : 'a.(?default : 'a?, ['a]) -> 'a,
  length : 'a.(['a]) -> int,
  make : 'a.(int, 'a) -> ['a],
  map : 'a.'b.(((c) -> 'b), ['a]) -> ['b]
  .{right : 'c.'d.(((c) -> 'd), ['c]) -> ['d]
},
  mapi : 'a.'b.(((int, 'a) -> 'b), ['a]) -> ['b],
  mem : 'a.('a, ['a]) -> bool,
  nth : 'a.(?default : 'a?, ['a], int) -> 'a,
  pick : 'a.(?default : 'a?, ['a]) -> 'a,
  prefix : 'a.(int, ['a]) -> ['a],
  remove : 'a.('a, ['a]) -> ['a],
  rev : 'a.(['a]) -> ['a],
  shuffle : 'a.(['a]) -> ['a],
  sort : 'a.(((c, 'a) -> int), ['a]) -> ['a]
  .{natural : 'b.(['b]) -> ['b]
},
  tl : 'a.(['a]) -> ['a]
```

Reduce memory usage

- Compiling phase
- Propagation of variables
- Only keep in memory what your script is using

Other improvements

Other improvements

- Autocue

Other improvements

- Autocue
- Cover manager, video canvas

Other improvements

- Autocue
- Cover manager, video canvas

```
let {px, vw, vh, rem, width, height} = video.canvas.virtual_10k.actual_720p
video.frame.width := width
video.frame.height := height
```

```
background =
  video.add_rectangle(
    color=0x333333,
    alpha=0.4,
    x=4609 @ px,
    y=234 @ px,
    width=5468 @ px,
    height=429 @ px,
    background
  )
```


Other improvements

- Autocue
- Cover manager, video canvas



Other improvements

- Autocue
- Cover manager, video canvas



Video Canvas and AI

Posted on February 10, 2024

Liquidsoap did not make it to FOSDEM this year, unfortunately. We had a nice example of advanced video use to present so here it is!

The code presented in this article is available here: <https://github.com/savonet/ai-radio>

The setup

We are looking at a cleaned-up version of a code that has been contributed by several

Other improvements

- Autocue
- Cover manager, video canvas
- Lots that are missing here!

Liquidsoap 2.3.x release?

Soon!



- Autocue backport
- Memory optimizations
- Performance and regression

Questions?