

# The Lost Art of Single File Ruby Programs

Christian Bäuerlein - @fabrik42

Hi!

# Hi! My name is Christian Bäuerlein



- Living in Frankfurt am Main
- [christianbaeuerlein.com](http://christianbaeuerlein.com)
- [@fabrik42 / @fabrik@mastodon.social](https://@fabrik42@mastodon.social)
- [github.com/fabrik42](https://github.com/fabrik42)
- CTO at [joki GmbH](https://joki.de) - a Deutsche Bahn company
- We build on-demand mobility platforms to enable public transport providers to roll their own DRT and autonomous services.

# ioki - Digital Public Transport!

Gold



DIGITAL PUBLIC TRANSPORT

We create smart mobility solutions for the future and work with transport service operators, local government bodies and companies in urban and rural regions alike. Based in Frankfurt, we are the technology partner that designs needs-based and modern on-demand mobility services to take us towards a more sustainable future. Our approach to mobility is holistic and data based. With our mobility analyses, traffic plannings and our operating system for digital mobility, we accompany mobility providers through the mobility transition. Our ecological and economically efficient solutions for a strong public transport make us the market leader in Europe.

[learn more](#)



# The Lost Art of Single File Ruby Programs

# The Lost Art of Single File Ruby Programs

A loose collection of Ruby fun facts and examples  
to organize your code in a single file.

# The Lost Art of Single File Ruby Programs

A loose collection of Ruby fun facts and examples  
to organize your code in a single file.

Let's have some fun with Ruby!

Are you ready? 

# A little Ruby history

# Ruby is a better Perl

Why the name *Ruby*?

Influenced by Perl, Matz wanted to use a jewel name for his new language, so he named Ruby after a colleague's birthstone.

Source: [The Ruby Language FAQ](#)

# Perl's legacy

Ruby took a lot of things from Perl.

Today we will learn about:

- Keywords
- Command line flags

# Code and tests in one file

# Ruby's pre-defined variables

There is **\$0**.

Contains the name of the file containing the Ruby script being executed.

Source: [Pre-defined variables and constants](#)

See also `$PROGRAM_NAME`

# Ruby's magic keywords

There is **\_\_FILE\_\_**.

The path to the current file.

Source: [ruby-doc.org](http://ruby-doc.org) Keywords

# The source file

```
$ cat greeter.rb

def greet(name)
  "Hello #{name}!"
end

# this will only run if the script was called directly
# not loaded or required
if __FILE__ == $0
  require "test/unit/assertions"
  include Test::Unit::Assertions

    assert_equal 'Hello Ruby', greet('Ruby'), "returns 'Hello Ruby! ''"
end
```

# When called directly

```
$ ruby greeter.rb
```

```
returns 'Hello Ruby!'. (Test::Unit::AssertionFailedError)
<"Hello Ruby"> expected but was
<"Hello Ruby!">.
```

```
diff:
```

```
- Hello Ruby
+ Hello Ruby!
?
```

# When required from another file

```
$ cat code_and_test_usage.rb  
require './greeter.rb'  
puts greet "Christian"
```

```
$ ruby code_and_test_usage.rb  
Hello Christian!
```

# The source file

```
$ cat greeter.rb

def greet(name)
  "Hello #{name}!"
end

# this will only run if the script was called directly
# not loaded or required
if __FILE__ == $0
  require "test/unit/assertions"
  include Test::Unit::Assertions

    assert_equal 'Hello Ruby', greet('Ruby'), "returns 'Hello Ruby! ''"
end
```

The **\_\_END\_\_** and **DATA** keywords

# Let's start with Perldata

Perl has two special literals:

**\_\_END\_\_**

Indicates the logical end of the script before the actual end of file.

**\_\_DATA\_\_**

A filehandle that points to everything that comes after **\_\_END\_\_**.

# The **\_\_END\_\_** and **DATA** keywords in Ruby

Denotes the end of the regular source code section of a program file.  
Lines below **\_\_END\_\_** will not be executed.

Those lines will be available via the special filehandle **DATA**.

Source: [Ruby-doc](#) - Class: [Object](#)

# Simple Example: A valid Ruby file

```
$ cat pets.rb
```

```
DATA.each_line do |line|
  puts line
end
```

```
--END--
```

```
Cats
```

```
Dogs
```

```
Hamsters
```

# Simple Example: A valid Ruby file

```
$ ruby pets.rb
```

Cats

Dogs

Hamsters

# Simple Example: A valid Ruby file

```
$ cat pets.rb
```

```
DATA.each_line do |line|
  puts line
end
```

```
--END--
```

```
Cats
```

```
Dogs
```

```
Hamsters
```

# ERB template and code in one file

```
require 'erb'

time = Time.now
renderer = ERB.new(DATA.read)
puts renderer.result()

__END__
The current time is <%= time %>.
```

# A web server in one file

# Sinatra has taken the stage

```
require 'rubygems'  
require 'sinatra'  
  
get '/' do  
  'Hello World'  
end
```

# Sinatra Templates: Uncool way

```
template :index do
  '%div.title Hello World!'
end
```

As documented in the 0.6 [README.rdoc](#) there was also a cool way to do it.

# Sinatra Templates: Cool way

```
get '/' do
  haml :index
end
```

```
use_in_file_templates!
```

```
--END--
@@ layout
%body
= yield
```

```
@@ index
%h1 Hello world!!!!
```

# Two templates, one file?

```
File.read(caller.first.split(":").first).split("__END__", 2).last
```

Source: [Mixing code and data in Ruby](#)

# Bundler Inline

# Bundler Fun Fact

- You don't need a **Gemfile** to use **bundler!**
- Useful for *Single File Programs™*
- Useful for scripts in your **/utils** folder that you only use once a year

Source: [How to use Bundler in a single-file Ruby script](#)

# Bundler Fun Fact

- You don't need a **Gemfile** to use **bundler!**
- Useful for *Single File Programs*™
- Useful for scripts in your **/utils** folder that you only use once a year - the **Trophy Scripts**

Source: [How to use Bundler in a single-file Ruby script](#)

# Bundler Fun Fact

- You don't need a **Gemfile** to use **bundler!**
- Useful for *Single File Programs*™
- Useful for scripts in your **/utils** folder that you only use once a year - the **Trophy Scripts**. Don't let them clutter your Gemfile!

Source: [How to use Bundler in a single-file Ruby script](#)

# Example

```
require 'bundler/inline'

gemfile do
  source 'https://rubygems.org'
  gem 'httparty'
end

puts HTTParty.get('https://www.boredapi.com/api/activity')
```

What happens: checks dependencies, installs dependencies, runs code.  
No Gemfile.lock will be written either.

# Example: Inline MiniTest suite

```
require 'bundler/inline'

gemfile do
  source 'https://rubygems.org'
  gem 'minitest', require: false
end

require 'minitest/autorun'

class MyTest < Minitest::Test
  def test_should_be_true
    assert_equal true, true
  end
end
```

# Example: Single File Rails App

- Used for reproducing bugs in the Rails issue tracker
- Multiple single file templates available, including Controllers, Models, Migrations, Storage, Background Jobs, etc

Source: [Rails Guides: Executable Test Cases](#)

The **BEGIN** and **END** keywords

# Yes, this is taken from Perl as well

**BEGIN** defines a block that is run before any other code in the current file.

Similarly **END** defines a block that is run after any other code.

Source: [Ruby Docs Miscellaneous Syntax](#)

See also `Kernel#at_exit`

Yes, this is taken from Perl **AWK** as well

Source: [AWK-ward Ruby](#)

# Yes, this is taken from Perl as well

**BEGIN** defines a block that is run before any other code in the current file.

Similarly **END** defines a block that is run after any other code.

Source: [Ruby Docs Miscellaneous Syntax](#)

See also `Kernel#at_exit`

# Example

```
END { puts 3 }
BEGIN { puts 1 }
puts 2
```

```
$ ruby begin.rb
```

```
1
2
3
```

# Introducing LRuby

# Logging Ruby

The Ruby alias for the forgetful scripter

Only Feature:

No more scrolling through your terminal...

Logs the output of a script to the script itself!

# Let's try this out!

```
$ cat examples/hello_world.rb
```

```
$ ruby examples/hello_world.rb
```



\*~/dev/ruby/iruby/-inferior-shell-1\*@iruby

```
1 puts "Hello world! It is #{Time.now.strftime('%H:%M:%S')}"
```

① - hello\_world.rb Ruby

edit ← unix | 2: 0 ← All →

## → examples git:(main)

**lruby** ② \* \*~/dev/ruby/lruby/-inferior-shell-1\* Shell :run  
-- INSERT --

terminal < utf-8 | 2:23 < [system] < All <

-- INSERT --

# Introducing LRuby

```
$ lruby examples/hello_world.rb
```

```
$ cat examples/hello_world.rb
```

\*~/dev/ruby/lruby/-inferior-shell-1\*@lruby

```
1 puts "Hello world! It is #{Time.now.strftime('%H:%M:%S')}"
```

① - hello\_world.rb Ruby

edit unix | 2: 0 All

# → examples git:(main) x

**lruby** ② \* \*~/dev/ruby/lruby/-inferior-shell-1\* Shell :run  
-- INSERT --

terminal < utf-8 | 2:25 < [system] < All <

-- INSERT --

# cURL

\*~/dev/ruby/iruby/-inferior-shell-1\*@iruby

```
1 puts `curl https://www.boredapi.com/api/activity`
```

① - curl\_example.rb Ruby

edit ◀ unix | 2: 0 ◀ All ▶

## → examples git:(main)

**lruby** ② ➔ \* \*~/dev/ruby/lruby/-inferior-shell-1\* ➔ Shell ➔ :run ➔

terminal < utf-8 | 2:23 < [system] < All <

# How does it work?

```
$ which lruby
```

```
lruby: aliased to  
ruby -r ~/single-file-ruby-programs/lruby.rb
```

# Require files via command line flag

**ruby -r [filename]**

Causes Ruby to load the file using **require**.

Source: [Ruby Docs Command line Options](#)

# LRuby code

```
BEGIN {
    require 'stringio'
    $stdout = StringIO.new
}

END {
    output = $stdout.string

    end_marker = '__END__'
    code, data = File.read($0).split(end_marker)

    time = Time.now.strftime('%Y-%m-%dT%H:%M:%S%z')
    headline = "----- [#{time}] RESULTS -----"
    new_data = [data, headline, output].join("\n")
    File.write($0, [code, new_data].join("#{end_marker}"))
    STDOUT.puts output
}
```

```
BEGIN {
    require 'stringio'
    $stdout = StringIO.new
}

END {
    output = $stdout.string

    end_marker = '__END__'
    code, data = File.read($0).split(end_marker)

    time = Time.now.strftime('%Y-%m-%dT%H:%M:%S%z')
    headline = "----- [#{time}] RESULTS -----"
    new_data = [data, headline, output].join("\n")
    File.write($0, [code, new_data].join("#{end_marker}"))
    STDOUT.puts output
}
```

```
BEGIN {
    require 'stringio'
    $stdout = StringIO.new

END {
    output = $stdout.string

    end_marker = '__END__'
    code, data = File.read($0).split(end_marker)

    time = Time.now.strftime('%Y-%m-%dT%H:%M:%S%z')
    headline = "----- [#{time}] RESULTS -----"
    new_data = [data, headline, output].join("\n")
    File.write($0, [code, new_data].join("#{end_marker}"))
    STDOUT.puts output
}
```

# Finally: Fire and forget!

# The Garbage flag

# Aaaaand back to Perl

**perl -x**

Leading garbage will be discarded until the first line

that starts with **#!** and contains the string "**perl**".

Source: [perlrun - perldoc.perl.org](#)

# Aaaaand back to Perl

**perl -x**

Leading **garbage** will be discarded until the first line

that starts with **#!** and contains the string "**perl**".

Source: [perlrun - perldoc.perl.org](#)

# But... why?

**perl -x**

Tells Perl that the program is embedded in a larger chunk of unrelated text, such as in a ...

Source: [perlrun - perldoc.perl.org](#)

# But... why?

**perl -x**

Tells Perl that the program is embedded in a larger chunk of unrelated text, such as in a [mail message](#).

Source: [perlrun - perldoc.perl.org](#)

# And in Ruby?

**ruby -x**

Tells Ruby that the script is embedded in a message.

Leading garbage will be discarded until the first line

that starts with "#!" and contains string "ruby".

Source: [Ruby Docs Command line Options](#)

# And in Ruby?

**ruby -x**

Tells Ruby that the script is embedded in a message.

Leading **garbage** will be discarded until the first line

that starts with "**#!**" and contains string "**ruby**".

Source: [Ruby Docs Command line Options](#)

# Example: A valid Ruby program

Hello dear friend,  
this is a message from the internet.  
Please execute it with your Ruby interpreter.

```
#! hahaha this is ruby now  
puts "Hello World"  
__END__
```

Thanks, a random stranger

```
$ ruby -x email.eml  
Hello World
```

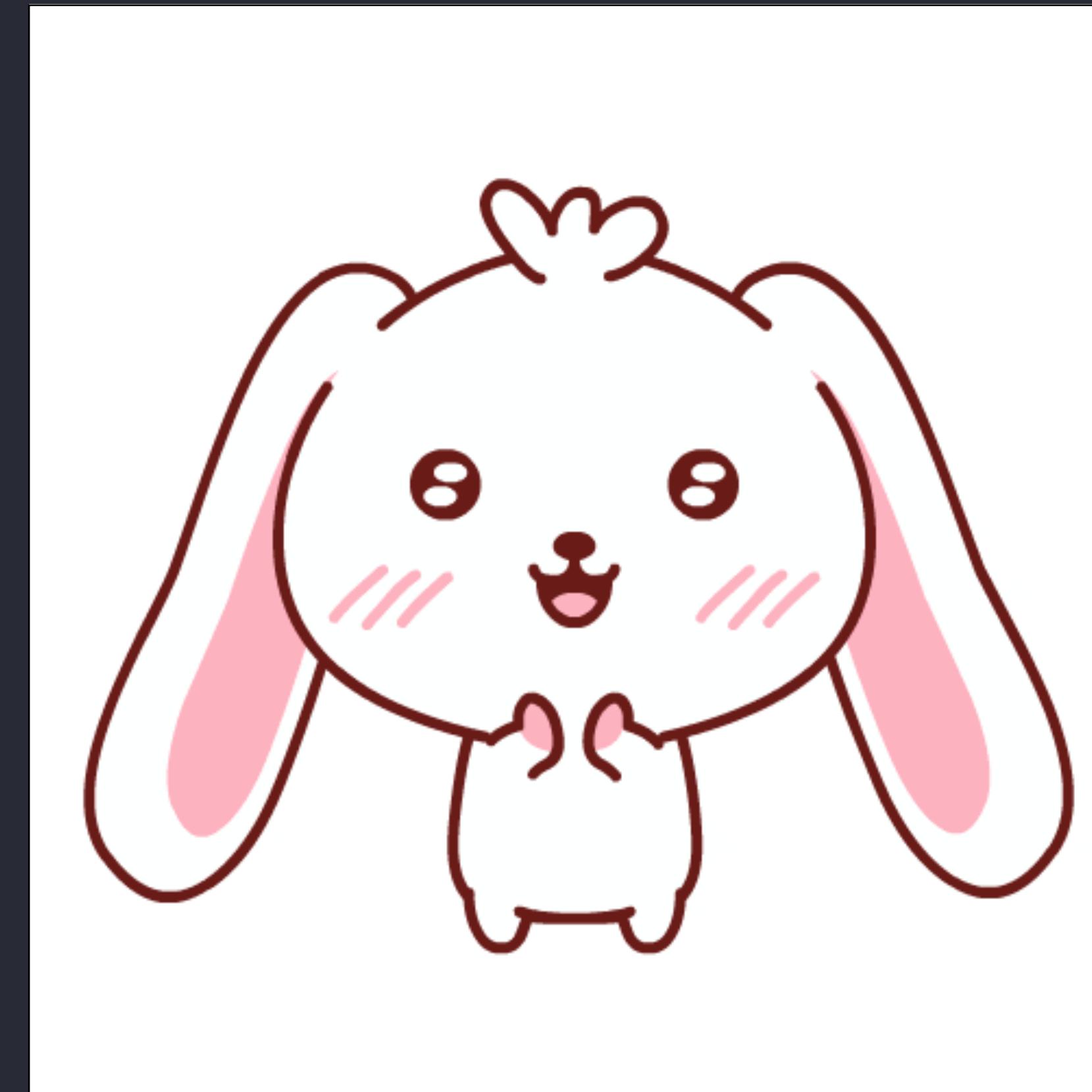
# A self-animating GIF

# A self-animating GIF?

This is not an animated GIF,  
but a GIF that animates itself.

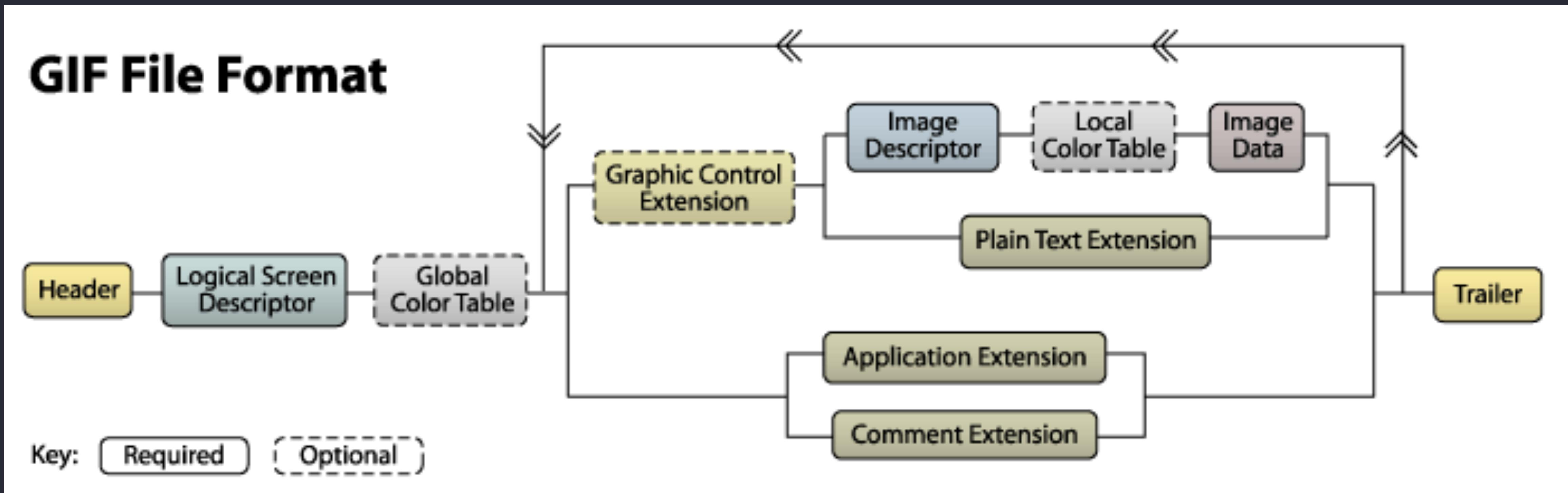


# Let's talk about GIFs



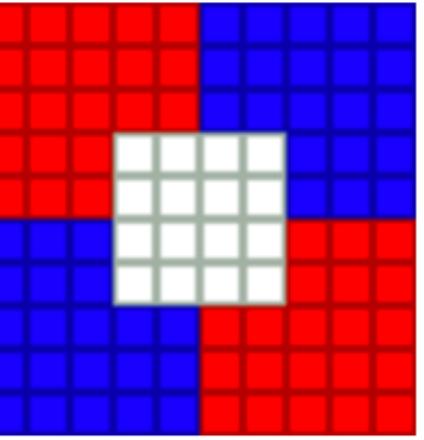
Source: "Happy Cheering" by Bluesbear

# A GIF file consists of blocks



Source: [What's in a GIF](#)

# Example

Enlarged	Bytes																																																																								
 (100x100)	<table><tbody><tr><td>47</td><td>49</td><td>46</td><td>38</td><td>39</td><td>61</td><td>0A</td><td>00</td><td>0A</td><td>00</td><td>91</td><td>00</td><td>00</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>00</td><td>00</td><td>00</td><td>00</td><td>FF</td><td>00</td><td>00</td></tr><tr><td>00</td><td>21</td><td>F9</td><td>04</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td>2C</td><td>00</td><td>00</td><td>00</td><td>00</td><td>0A</td><td>00</td><td>0A</td><td>00</td><td>00</td><td>02</td><td>16</td><td>8C</td><td>2D</td><td>99</td></tr><tr><td>87</td><td>2A</td><td>1C</td><td>DC</td><td>33</td><td>A0</td><td>02</td><td>75</td><td>EC</td><td>95</td><td>FA</td><td>A8</td><td>DE</td><td>60</td><td>8C</td><td>04</td><td>91</td><td>4C</td><td>01</td><td>00</td><td>3B</td><td></td><td></td><td></td></tr></tbody></table>	47	49	46	38	39	61	0A	00	0A	00	91	00	00	FF	FF	FF	FF	00	00	00	00	FF	00	00	00	21	F9	04	00	00	00	00	00	2C	00	00	00	00	0A	00	0A	00	00	02	16	8C	2D	99	87	2A	1C	DC	33	A0	02	75	EC	95	FA	A8	DE	60	8C	04	91	4C	01	00	3B			
47	49	46	38	39	61	0A	00	0A	00	91	00	00	FF	FF	FF	FF	00	00	00	00	FF	00	00																																																		
00	21	F9	04	00	00	00	00	00	2C	00	00	00	00	0A	00	0A	00	00	02	16	8C	2D	99																																																		
87	2A	1C	DC	33	A0	02	75	EC	95	FA	A8	DE	60	8C	04	91	4C	01	00	3B																																																					

Source: [What's in a GIF](#)

# Terminator Byte

The trailer block indicates when you've reached the end of the file.

It is always a byte with a value of **3B**.

btw: The hexadecimal value **3B** is **59** in decimal.

The ascii value **59** is a semicolon.

Source: [What's in a GIF](#)

# What we learned so far

- GIFs are nice
- GIFs always end with the same terminator byte
- Ruby is nice
- Ruby can start at a defined line (ignoring leading *garbage*)
- Nice.

# A self-animating GIF!

# Demo time!

Let's check out the **rbgif.gif** source code together!

# What will happen now

- A loop in my shell will keep calling the ruby script (gif file)
- The file will rewrite itself (the upper part aka the gif part)
- We will watch the progress in the browser

# DEMO

fabrik42/single-file-ruby-progra X fabrik42/lruby: Logging Ruby - T Euruko 2023

file:///Users/fabrik42/Desktop/single-file-ruby-programs/rbgif/render.html

# I am a gif and a Ruby script!

2023-09-21 15:58:29 +0200

.....

while (ruby) ..ev/ruby/lruby (-... Ruby

```

1 GIF89a[356^B\310^@]367^@~^D\202z^L\206v^T\212x^P\210r^[\215n\$222j+\225p\220f4\232h0\230b<\236^D\2>
2 \353\365^P\340\360^F\364\371^H\360\370^@]377\301\203\301\306\214\306\311\224\311\315\233\315\322^>
3 ^]J\264\250\321\243H\223*]312\264\251\323\247P\243J\235J\265\252\325\253X\263j\335\312\265\253\327\2>
4 \204\337\221\342\361^W\350\240\205^&^377\232\250A\214:\272\301\250\206B\201\246\231\220\312\204|pr
5 \353\247\2636D\244BG\336)320^E\317\215^P\263\232(k^A\213^W\224\300^F\$230RJP^G\315^&^364\303s^]*t\3>
6 ^B$^@]2528\220\301^@]372^P\254B^_213^Gs\240r*\204s\203\313v[^P\264\322^V\310\301^]@]262\261A\314^>
7 m\300(\200T\$^B\315\343^]R\346@]362\206\360Ax^H[\240/y^Y\2408\236^Z\341Z\200\263xe[\240\360\302\246>
8 T\347sL\220P\302\346\344\371\267\320\343\251KN9B\307\246+o^DA\353\254Av^NT\323w^@!\302u\234\267\210>
9 ^B\215K^G\321@^H\360\367\265\242iby^Ei\236@]377
10 49E] }232\210^[\267^ZU\220\331i^B{^G^A\301s^5\220^K0^My^C^Y\355, EMx\316^B\214c^C^T\315\346\301i\3>
11 "202^T\301I^E\242^A^L\310/\366\371\210ye\264\234*\361\327"6
12 d\211\244#\#h\342\330\220#\331\221Q\261$]210\340\356^220\364m\362!316\310\346\250^Y\245D\276\210\22>
13 \251&\312\331\327^_320\225\261\234\352\251\2674[\312\246^B^S\225\232\310^P^Q^Y\263X^200fxEH^G\324&>
14 \201a^L\231\360^C^P^T\250\261\222^]_^Q"\334\356^P\327\270^=H^F\236\203\317\203^X\326\254\236\314Sp\>
15 W\264\214:\363\205e^Mh\202\324\273\367^H^F\336+Z"\316y\324\332\365\330s\354^]^^|\267X\317^T}N v\365>
16 \337\351^V^PkQ\327of\222^?/\263^?a\347^?^L\261^AE\360^332B3\222\366M\211\245^P^&^>\200\201\200\200\2>
17 \225}PEM^Z^P^D\330\321.\$]240\202K\364e^K\221\2032\210)P\240KS\245\203\220^P^H]\240-\361\302q$fe^Zp^A>
18 ^A^A^U^@]217^G0^Q\355\370\216\3618^Q^E\340\216^S^@^A^A^@]217\366\210\217\363\250^@]331\217#2>
19 \200^C>Y^A^A^P^C\>231^B\221y^P\246\211\232\251\271\232 A^@]240\233/p\233^W\321\232\257^Y\233^V0>
20 @]236\345i^C\352x^P^K\300\241^LP^P^@]360^B\>311\221^H\261\241\345i\226^_j^P^D^@]2250\200^B'\300^Be\35>
21 \236^!^@]306\212\254Q)\245^M^Q^@G\311^C<\260^C\326J\255\325\232^C3\300\253^U!\255=\200\255\325j\255>
22 ^A^B\360\233Q\250^Ca\261=^P\246^B\261\245=^P\247^B^Q\26140\261^U\213\225^X\333\261=^32>
0\254$]261\264^E\373^P&\340\2233\240^PG*^C^U\321\263?k\262^Ga\2636\200\263:^K^S0\213^P3>
^A\225\244\373\266?371\243
^EQ\271>y\271u\253\271\234+\247{\333\233\223[^P\270K\270\243\233\267\303\333\267>371\2>
275^E^A^A0\273\360\223S{^Q\336^K\276?^Y\247G\332^C\337\233^P\311\212\260^VA\276\275>
302\234^PW\314\307^C\261\307H\354\307\351\352\306\377\201\310\273\321\276^Q|^P^R^P>
P\246\374\312z\225\252\274\277H\331^P\242\254^P\227\227\231\334^C\217l^P\310<\311@9>
17>)\317^V\220\272>\231\315v^Z\225\201\274\312\364\274^P\366\234\305\264\274^Q]\334^Pg\>
^@i\275\326^U\321\325\350;^PV\355\223X=^PZ\275\322\321\274^P0\335^CC\235^PR\375\232^R\3>
54\227^K^A\22519^R\242\315\320^B\354^Q`\335\306\345\371^B\367<\3104;\310]\211\333^K\261>
54?251^C^D-^P5
\340\211\340D\355\332>i^B^@]@^X\272^C^L^P^K^Q\374=^P\365\355\223\367]^P\371\275^C\2>
\344^P^Q\347^&^225K\216\337\347+^Pz\376\343\203\256\223j\377\337^&^333^C\317l^Pn\256>
\216\225\230~^P^A^@]334?yCl^>\310\370^&^Q\335\256\234\351\236\332?>\331\356\305\216\347 A>
\346\261^L\340NM^PB\357\224\365\316^C\276~\231\365\336\327\230j\24240\237X\271^Ct\217^>
64\220\263
2\313P\244^M#\352^P\242^@^H\250\314\304\347\242\213^H\304^K\262\316\301^&7S@]244^F^&^L\241!^M^Z1!\265\202\352\2>
7e\2305jY3\210\306!e\217!^z\246\377\235\256\260\240\251!^352\270j\212\267\346\232X^Bl>
\201^V^A\300j^Z\246^Q\377^F&\304\201\313k^F\213\323\203^Ru\206\200^F\224^H\363.\370\2>
\350^F^EW\242\310u\265+^231^D\253^F[\2153\210^&^Sj^A\262\226\265\250\377~^Bkf\304zK>
^\\310j]\233\304\201\370\354.6h\313^K&\333\231\327\272T\266\264\265-n\345\302\200\342\>
12\245\262\225\261^&^H\260@]3059^@]261^C\336f\330-3\244\307Y2q^ap\254c\350^E\300^U^Q\>
\300^L\324\300^M\344\300^N\364\300^D\301^P^T\301^0$]301^R4\301^SD\301^TT\301^Ud\301^V>
uby!
cii
g_literal: true
er/inline'
s://rubygems.org'
vg'

#!/let's go ruby!\n"

```

edit no-conversion | 1: 6 Top

# A self-animating GIF!



# Summary

# Summary: Single File Ruby Programs

## What we learned

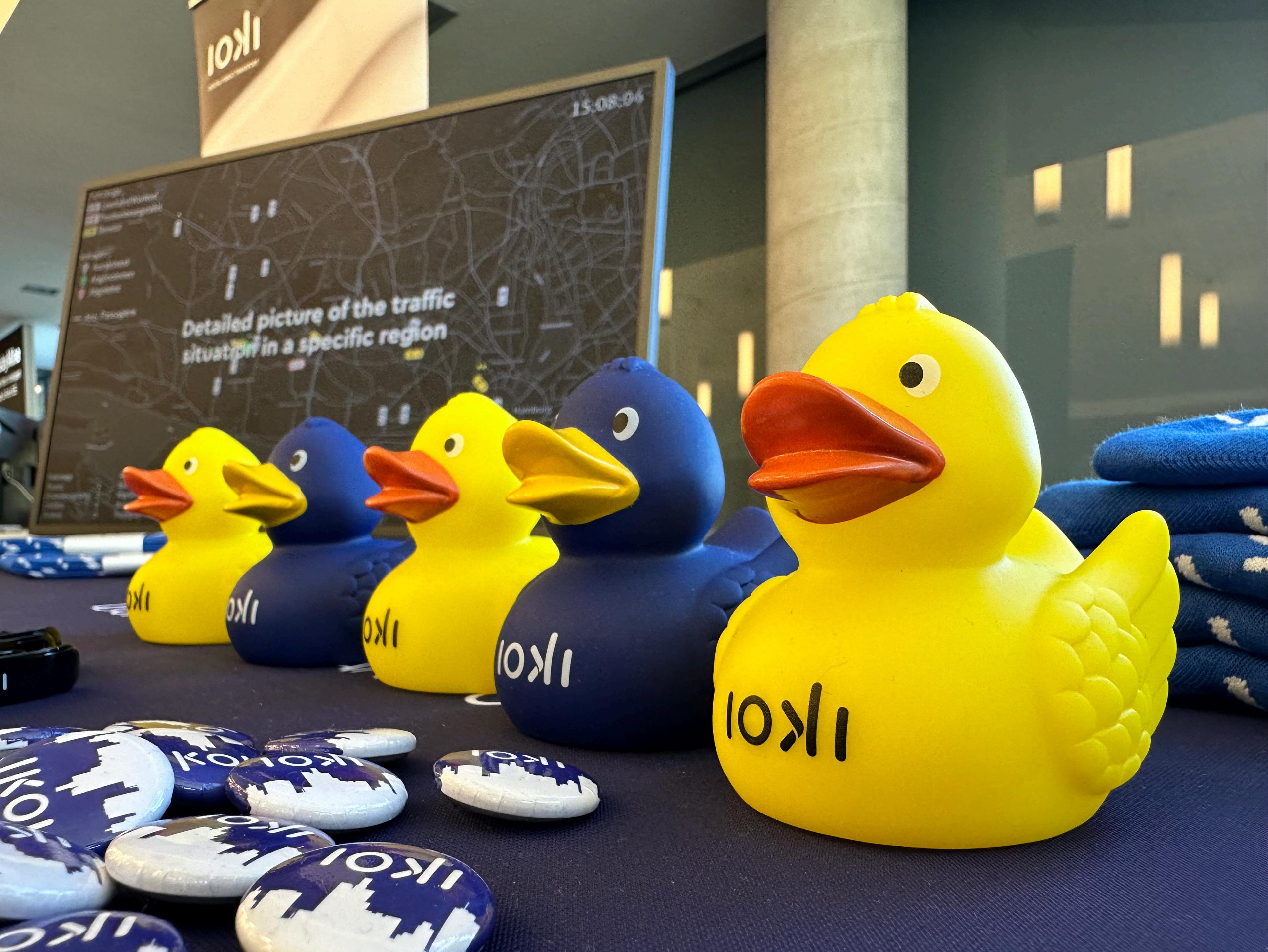
- Code & Tests
- Dependencies & Code
- Data & Code
- Code & Output
- Graphics & Code
- **Try it out for fun and profit!**

# Thanks!

Christian Bäuerlein - @fabrik42

<https://ioki.engineering>





# Single File Ruby Programs

- Code and Slides at  
<https://github.com/fabrik42/single-file-ruby-programs>
- LRuby repository  
<https://github.com/fabrik42/lruby>
- Learn more about ioki  
<https://ioki.engineering>

