Coroutines in PHP

Aurelijus Banelis

VilniusPHP 0x4F
2019-06-06

Aurelijus Banelis

Backend/DevOps

aurelijus.banelis.lt
aurelijus@banelis.lt

```
PGP    0x320205E7539B6203
130D C446 1F1A 2E50 D6E3
3DA8 3202 05E7 539B 6203
```

.NFQ

AKADEMIJA .NFQ

Auginte

home24

Coroutines in PHP

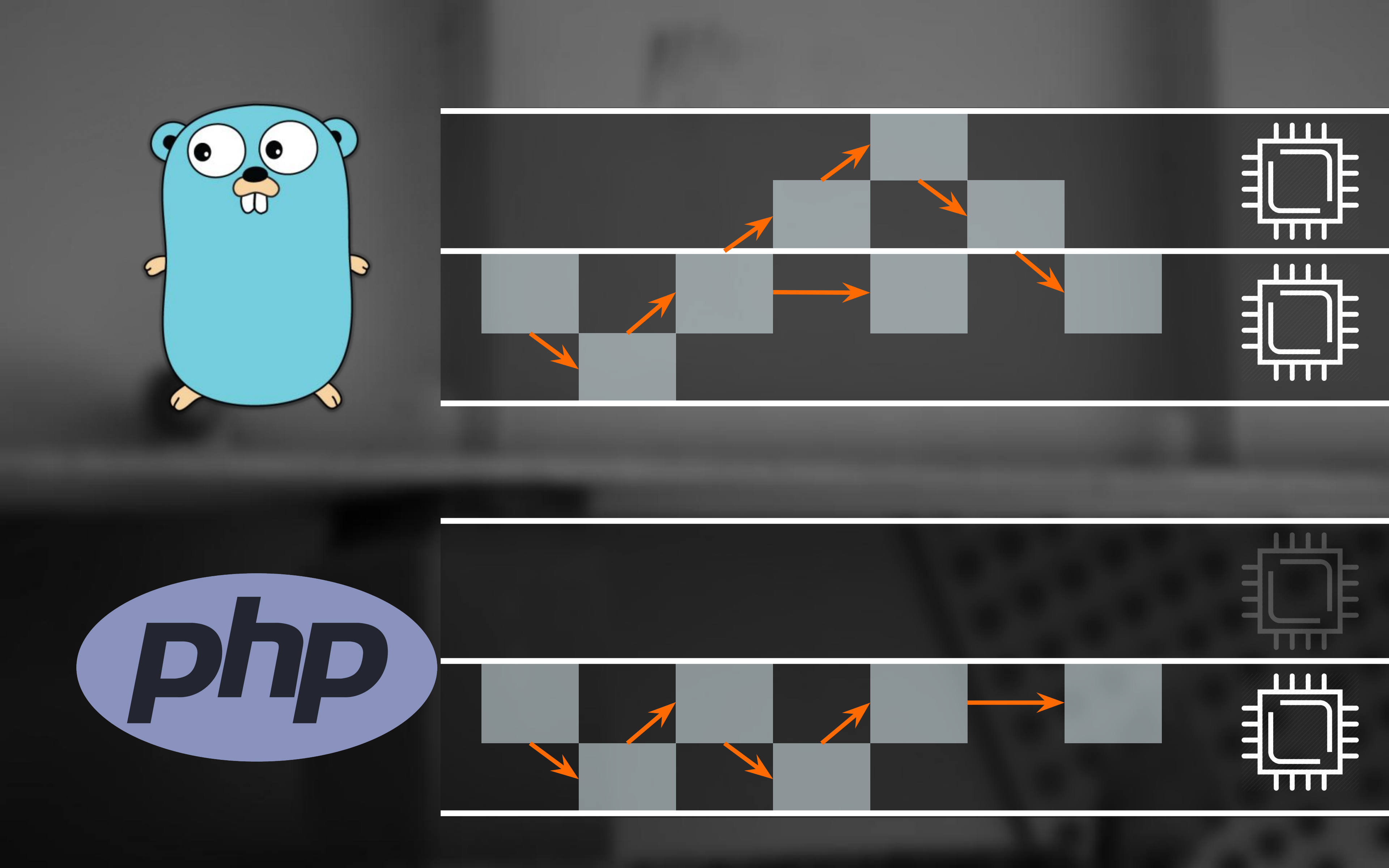
...and other asynchronous stuff

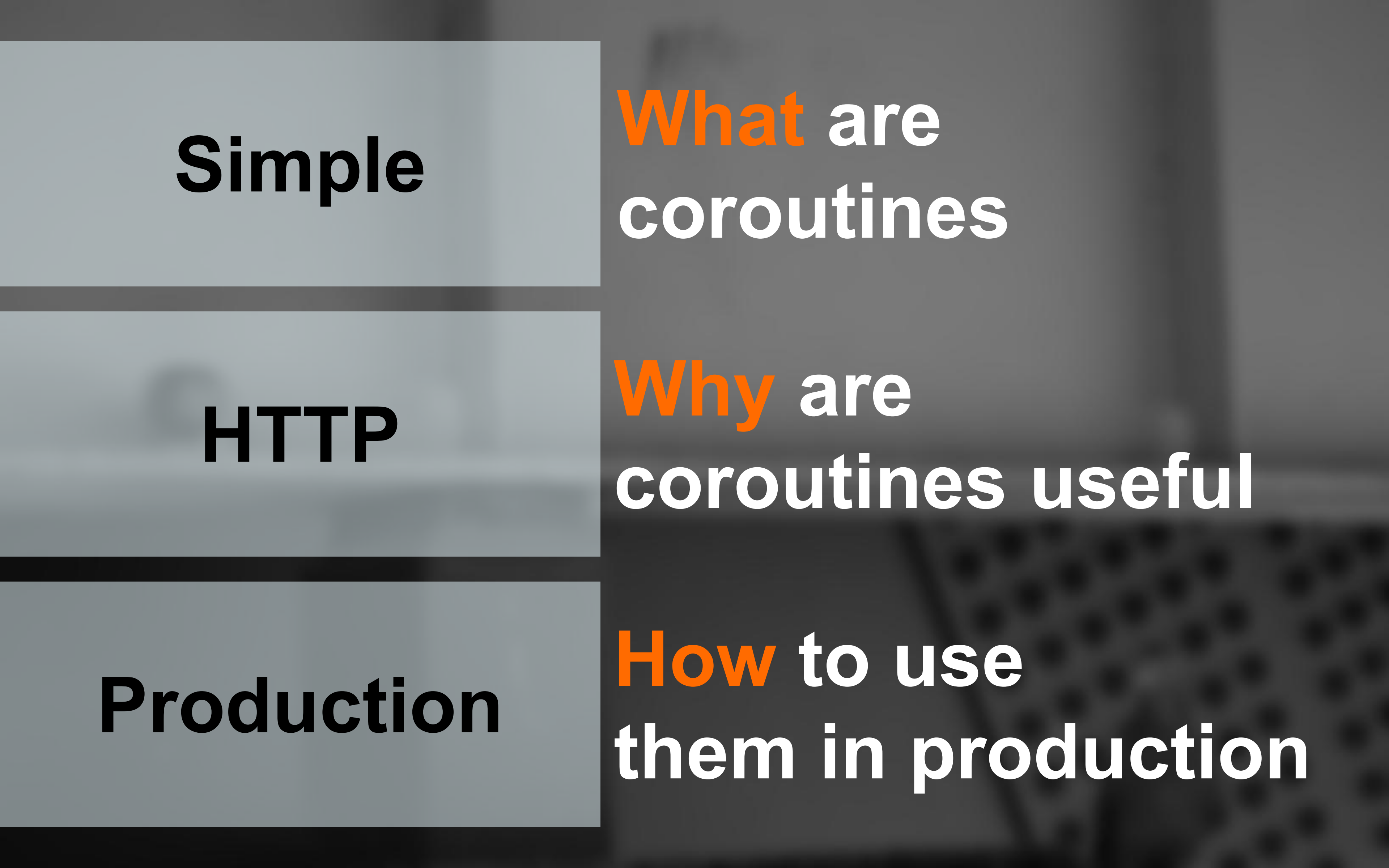
Goroutines

Coroutines

A goroutine is a lightweight thread managed by the Go runtime

Coroutines are computer program components that generalize subroutines for non-preemptive multitasking, by allowing execution to be suspended and resumed.


php



Simple

What are coroutines

HTTP

Why are coroutines useful

Production

How to use them in production

Simple

What are coroutines

HTTP

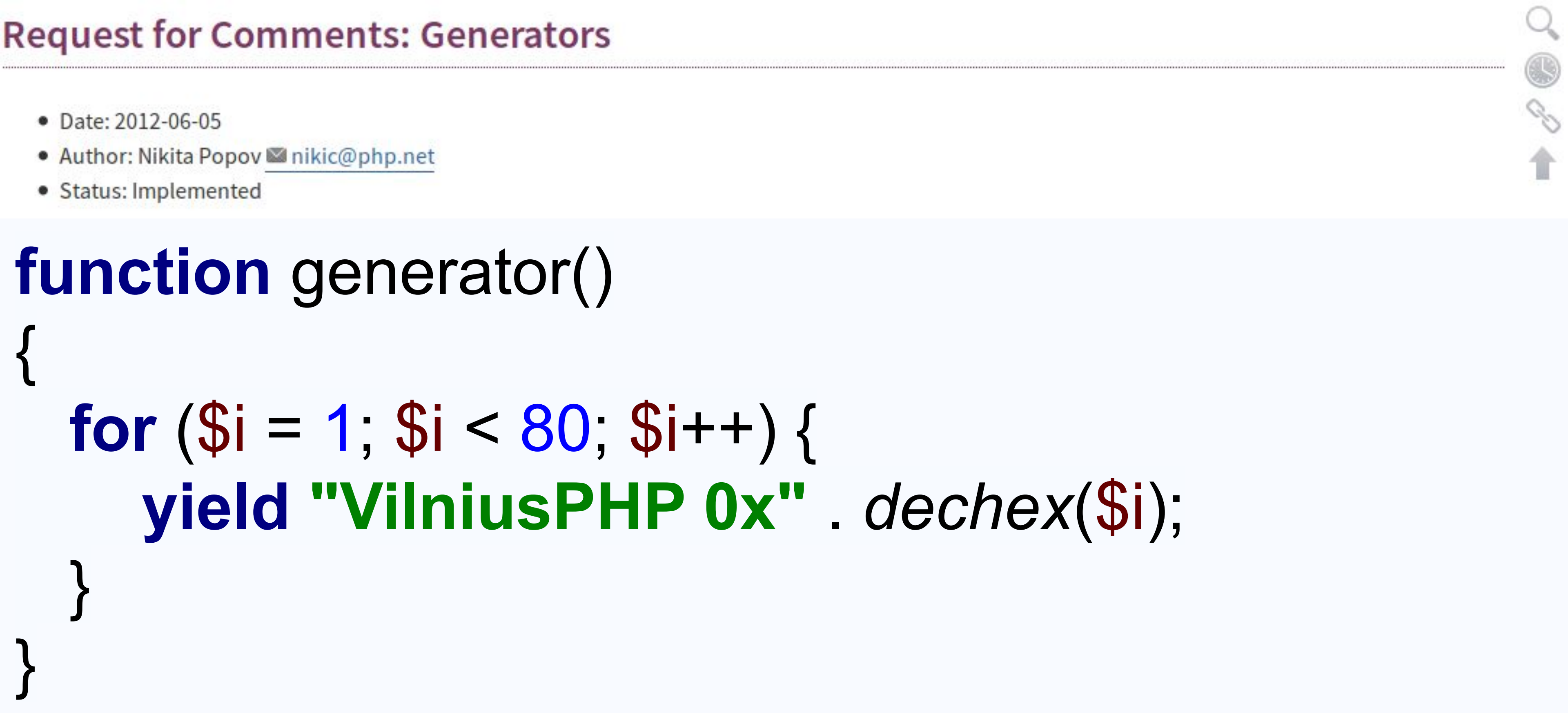
Why are coroutines useful

Production

How to use them in production

Request for Comments: Generators

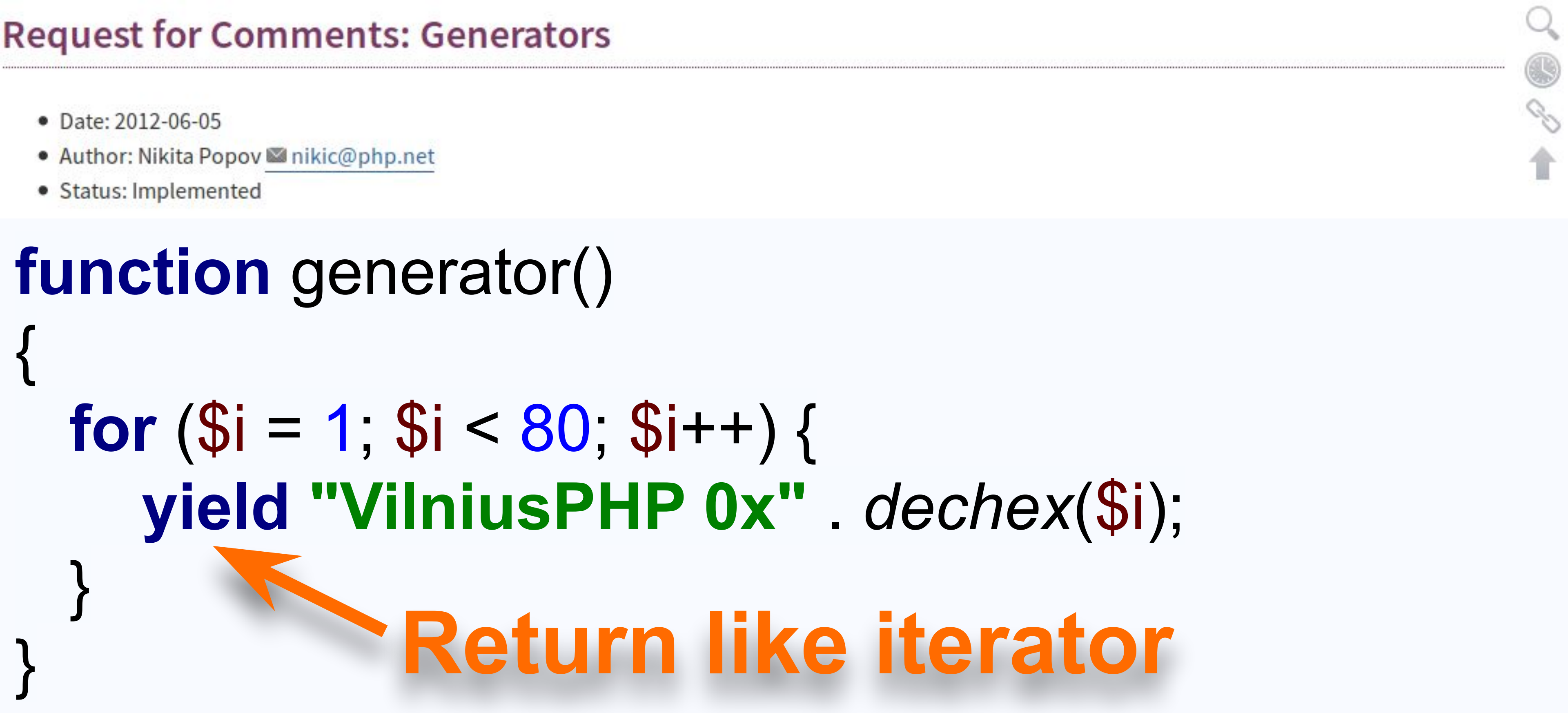
- Date: 2012-06-05
- Author: Nikita Popov ✉ nikic@php.net
- Status: Implemented



Request for Comments: Generators

- Date: 2012-06-05
- Author: Nikita Popov nikic@php.net
- Status: Implemented

```
function generator()
{
   for ($i = 1; $i < 80; $i++) {
      yield "VilniusPHP 0x" . dechex($i);
   }
}
```

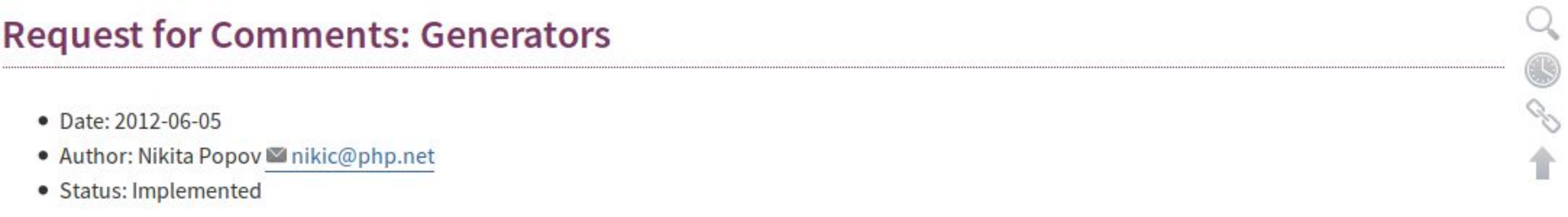


Request for Comments: Generators

- Date: 2012-06-05
- Author: Nikita Popov nikic@php.net
- Status: Implemented

```php
function generator()
{
   for ($i = 1; $i < 80; $i++) {
      yield "VilniusPHP 0x" . dechex($i);
   }
}
```

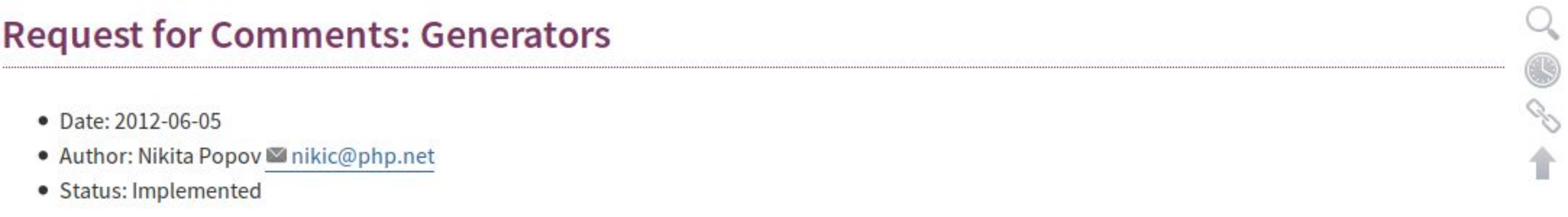
Return like iterator



Request for Comments: Generators

- Date: 2012-06-05
- Author: Nikita Popov nikic@php.net
- Status: Implemented

Generators can also be used the other way around,
i.e. instead of producing values they can also **consume** them.
When used in this way they are often referred to
as enhanced generators, reverse generators or **coroutines.**

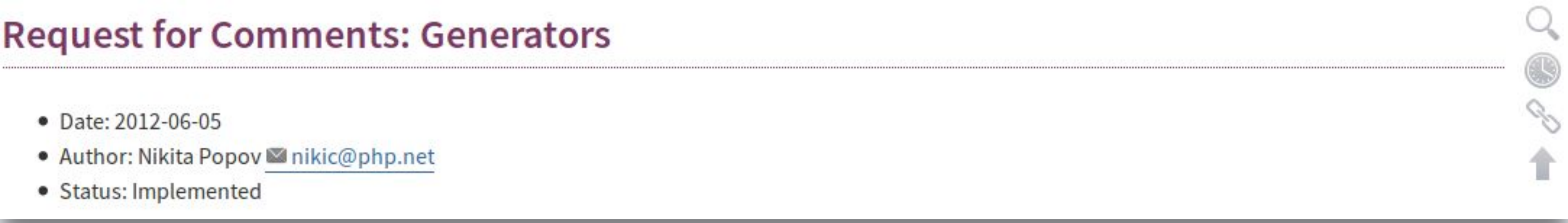


Request for Comments: Generators

- Date: 2012-06-05
- Author: Nikita Popov nikic@php.net
- Status: Implemented

```php
function receiver() {
  $data = (yield);
  print $data . " 0x4F\n";
}
receiver()->send("VilniusPHP");
```

Generators can also be used the other way around,
i.e. instead of producing values they can also **consume** them.
When used in this way they are often referred to
as enhanced generators, reverse generators or **coroutines.**



Request for Comments: Generators

- Date: 2012-06-05
- Author: Nikita Popov nikic@php.net
- Status: Implemented

```
function receiver() {
   $data = (yield);
   print $data . " 0x4F\n";
}
receiver()->send("VilniusPHP");
```

Consume

Generators can also be used the other way around,
i.e. instead of producing values they can also **consume** them.
When used in this way they are often referred to
as enhanced generators, reverse generators or **coroutines.**

Do you use yield?

Yield available since PHP 5.5

Coroutines were initially developed in the 1960's and then just sort of **died quietly**

Wonder why?

Let’s compare simple examples

Subroutines

```
Parsing started...
City: Vilnius
Language: PHP
Event: 0x4F
Finished...
```

Coroutines

```
Parsing started...
City: Vilnius
Parsing...
Language: PHP
Parsing...
Event: 0x4F
Finished...
```

```php
function formatter($city, $language, $event)
{
   print "City: $city\n";
   print "Language: $language\n";
   print "Event: $event\n";
}

function main()
{
   print "Parsing started...\n";
   formatter("Vilnius", "PHP", "0x4F");
   print "Finished...\n";
}

main();
```

Link to Gist

```
Parsing started...
City: Vilnius
Language: PHP
Event: 0x4F
Finished...
```

https://gist.github.com/aurelijusb/cacef3e4d57ca6e1772104e9a85079a9

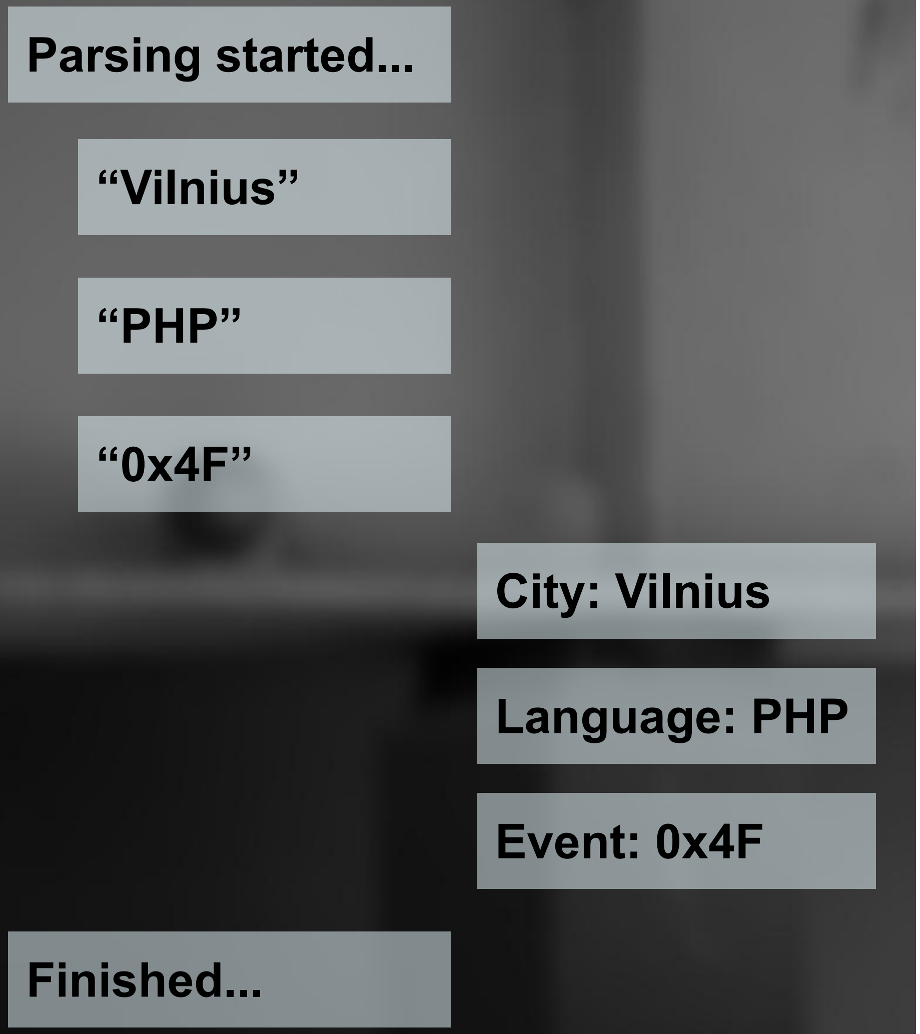
```php
function formatter()
{
   $city = (yield);
   print "City: $city\n";
   $language = (yield);
   print "Language: $language\n";
   $event = (yield);
   print "Event: $event\n";
}

function main()
{
   $p = formatter();
   print "Parsing started...\n";
   $p->send("Vilnius");
   print "Parsing...\n";
   $p->send("PHP");
   print "Parsing...\n";
   $p->send("0x4F");
   print "Finished...\n";
}

main();
```

Yield

```
Parsing started...
City: Vilnius
Parsing...
Language: PHP
Parsing...
Event: 0x4F
Finished...
```



Parsing started...

"Vilnius"

"PHP"

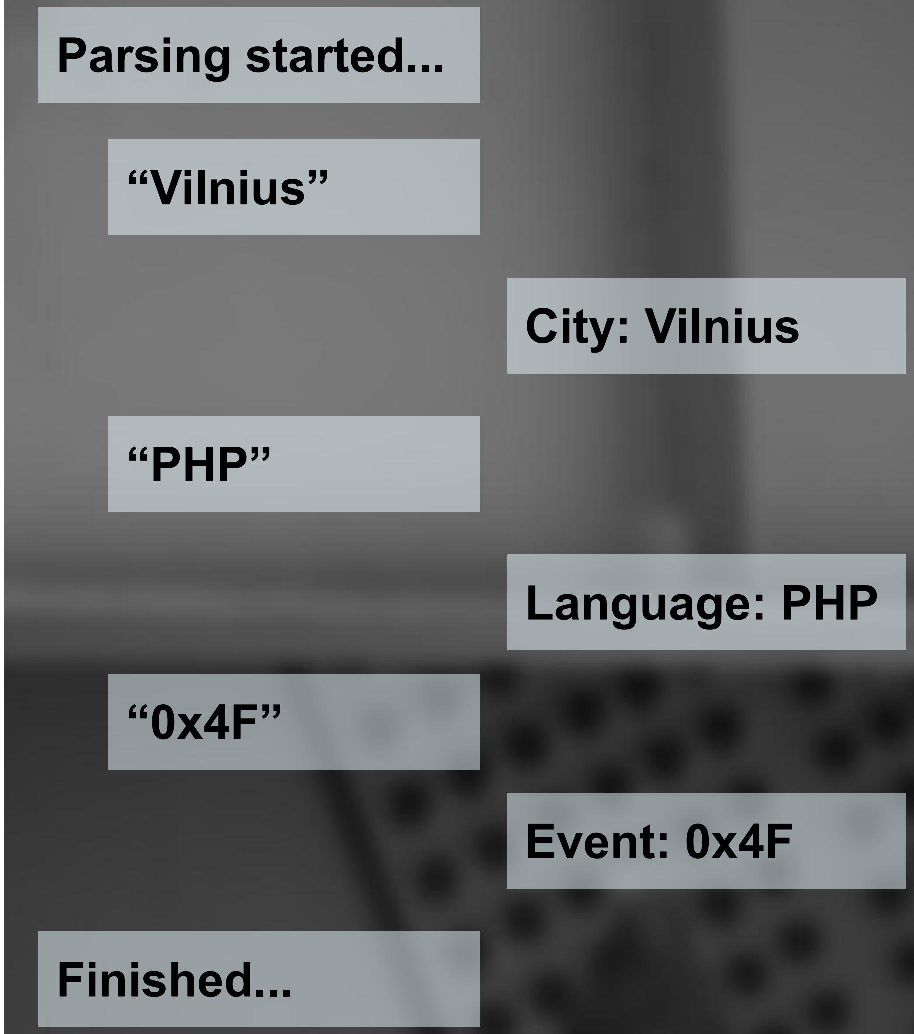
"0x4F"

City: Vilnius

Language: PHP

Event: 0x4F

Finished...



Parsing started...

"Vilnius"

City: Vilnius

"PHP"

Language: PHP

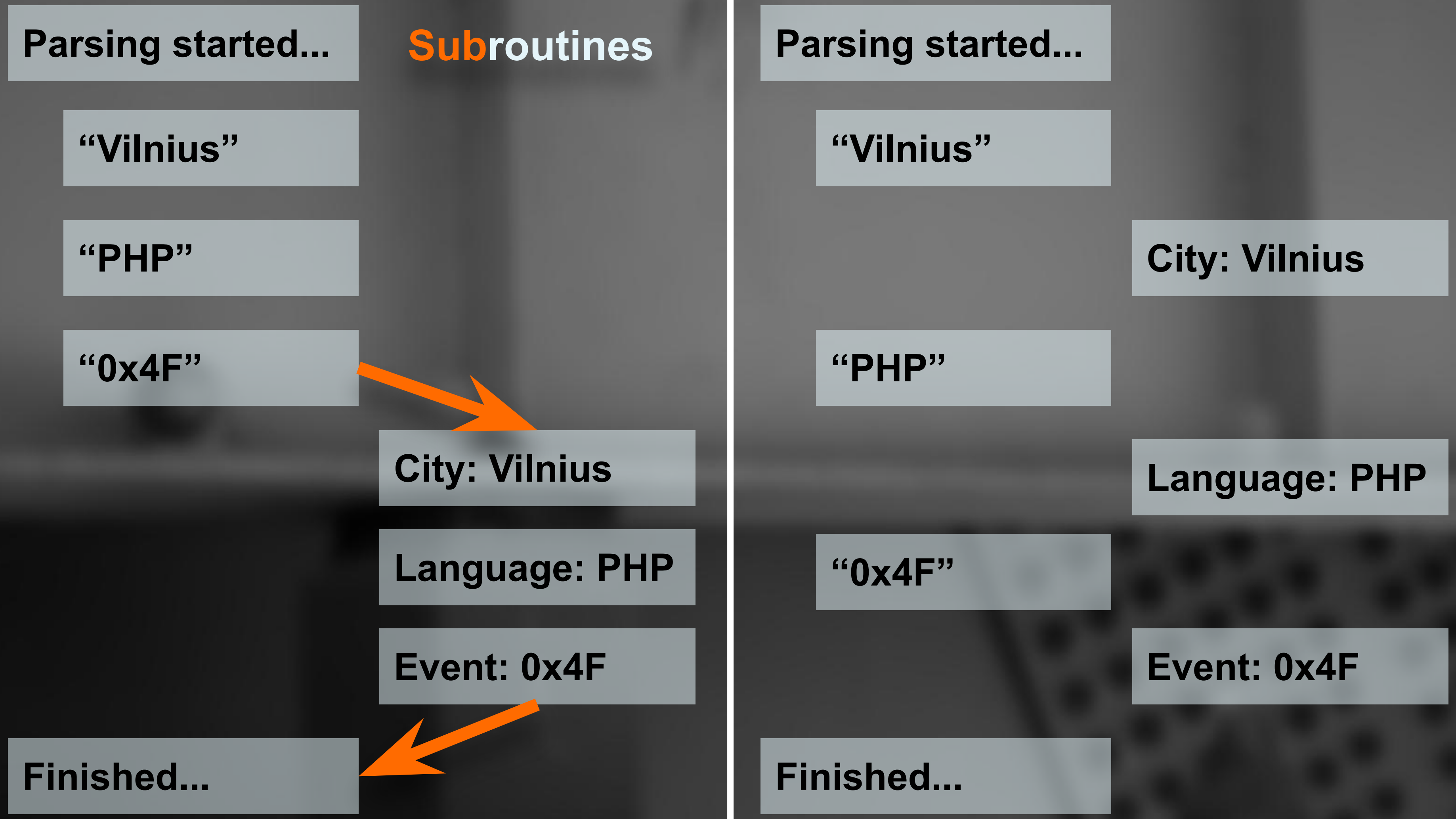
"0x4F"

Event: 0x4F

Finished...

https://gist.github.com/aurelijusb/cacef3e4d57ca6e1772104e9a85079a9

Subroutines



Parsing started...

“Vilnius”

“PHP”

“0x4F”

City: Vilnius

Language: PHP

Event: 0x4F

Finished...

Parsing started...

“Vilnius”

City: Vilnius

“PHP”

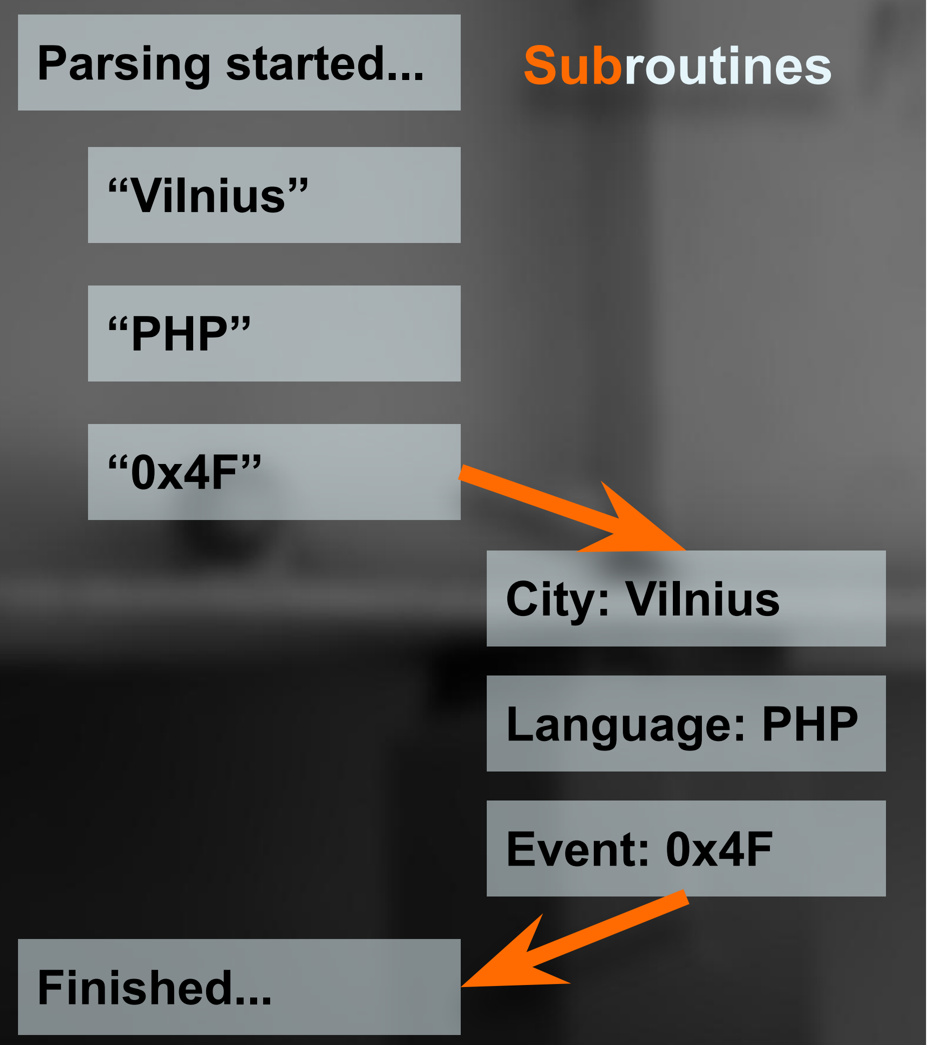
Language: PHP

“0x4F”

Event: 0x4F

Finished...

https://gist.github.com/aurelijusb/cacef3e4d57ca6e1772104e9a85079a9



Subroutines

Parsing started...

"Vilnius"

"PHP"

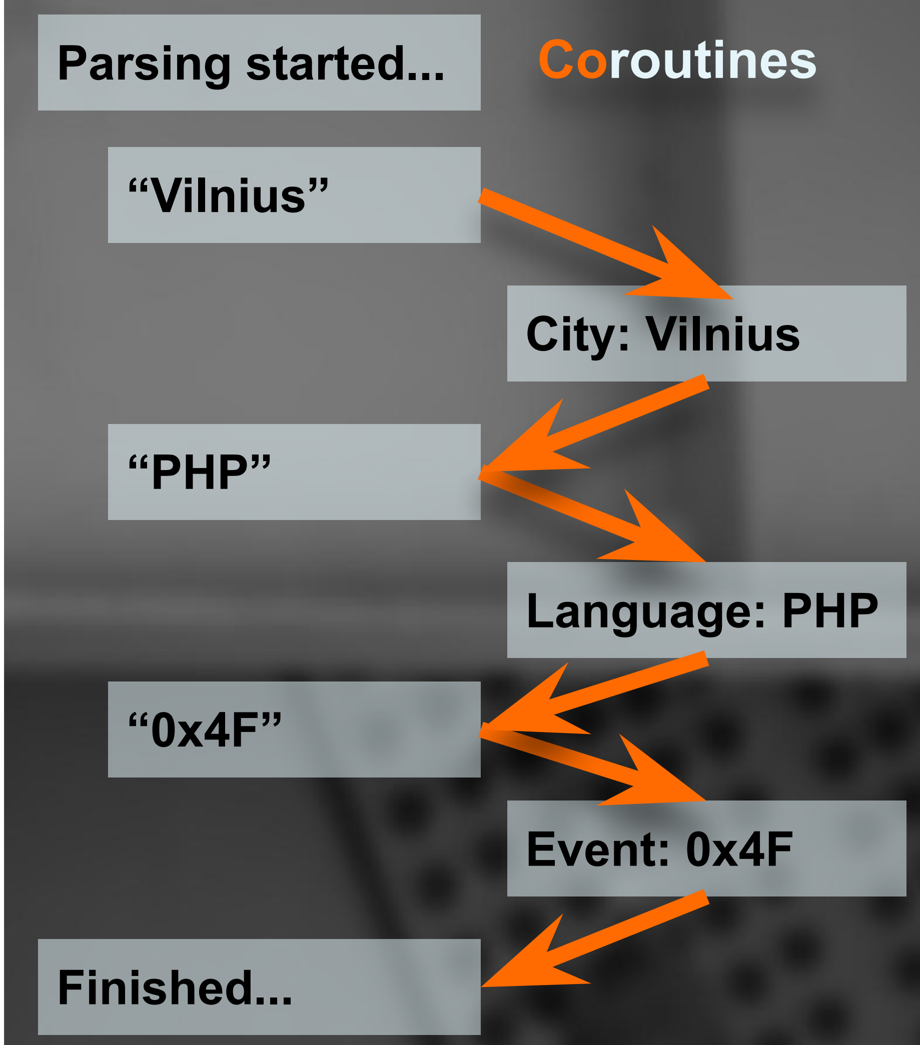
"0x4F"

City: Vilnius

Language: PHP

Event: 0x4F

Finished...



Coroutines

Parsing started...

"Vilnius"

City: Vilnius

"PHP"

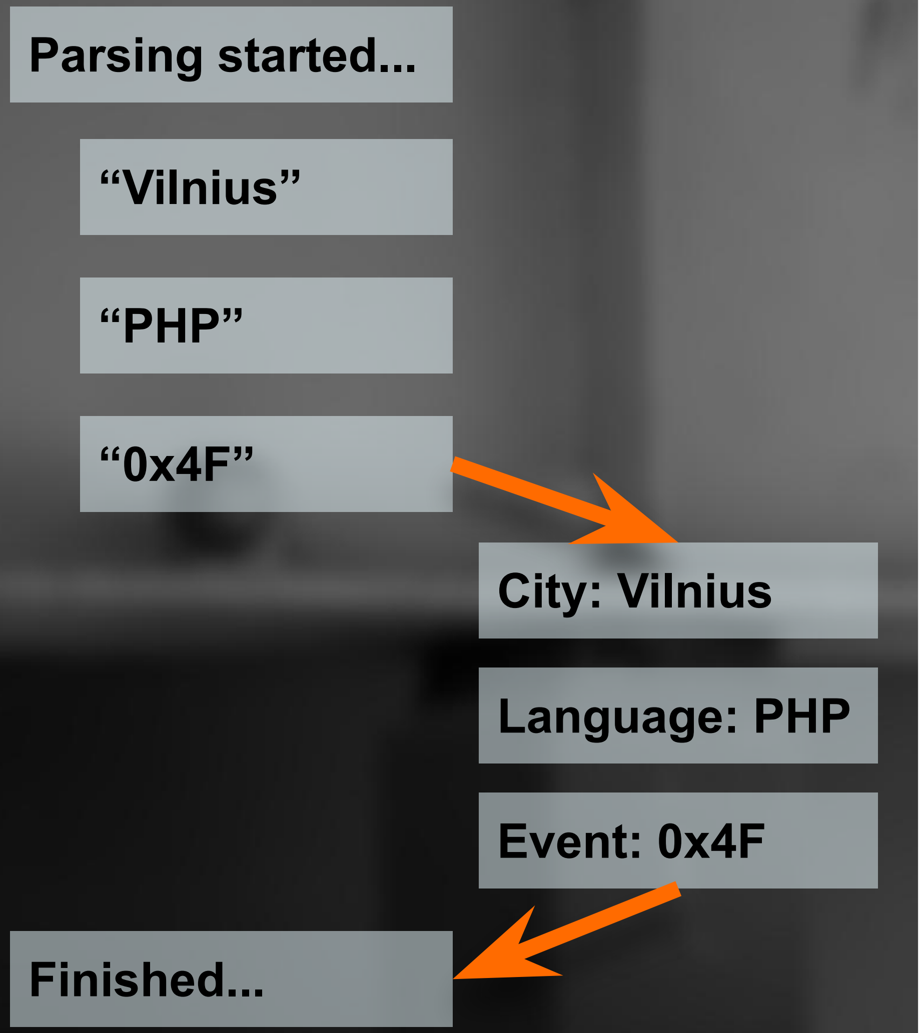
Language: PHP

"0x4F"

Event: 0x4F

Finished...

https://gist.github.com/aurelijusb/cacef3e4d57ca6e1772104e9a85079a9



Parsing started...

“Vilnius”

“PHP”

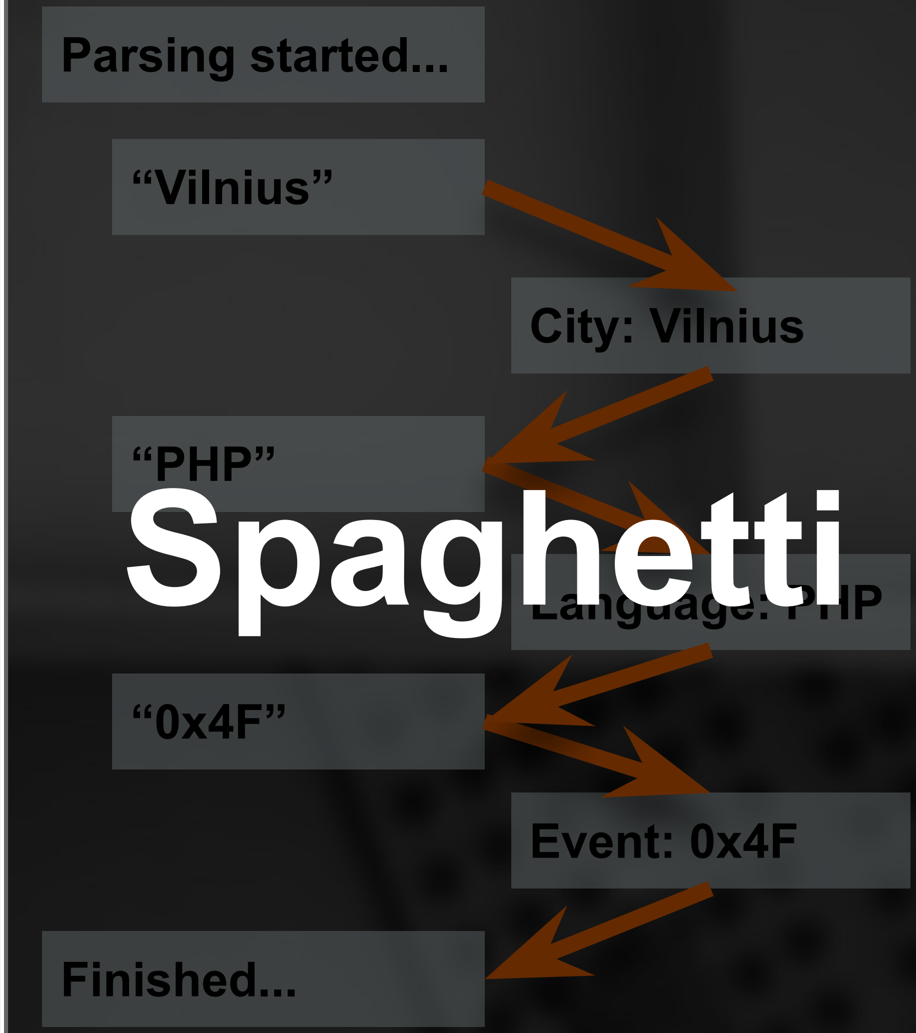
“0x4F”

City: Vilnius

Language: PHP

Event: 0x4F

Finished...



Parsing started...

“Vilnius”

City: Vilnius

“PHP”

Language: PHP

“0x4F”

Event: 0x4F

Finished...

Spaghetti

https://gist.github.com/aurelijusb/cacef3e4d57ca6e1772104e9a85079a9

Simple	Emit + receive in the middle of func
HTTP	Why are coroutines useful
Production	How to use them in production

Simple

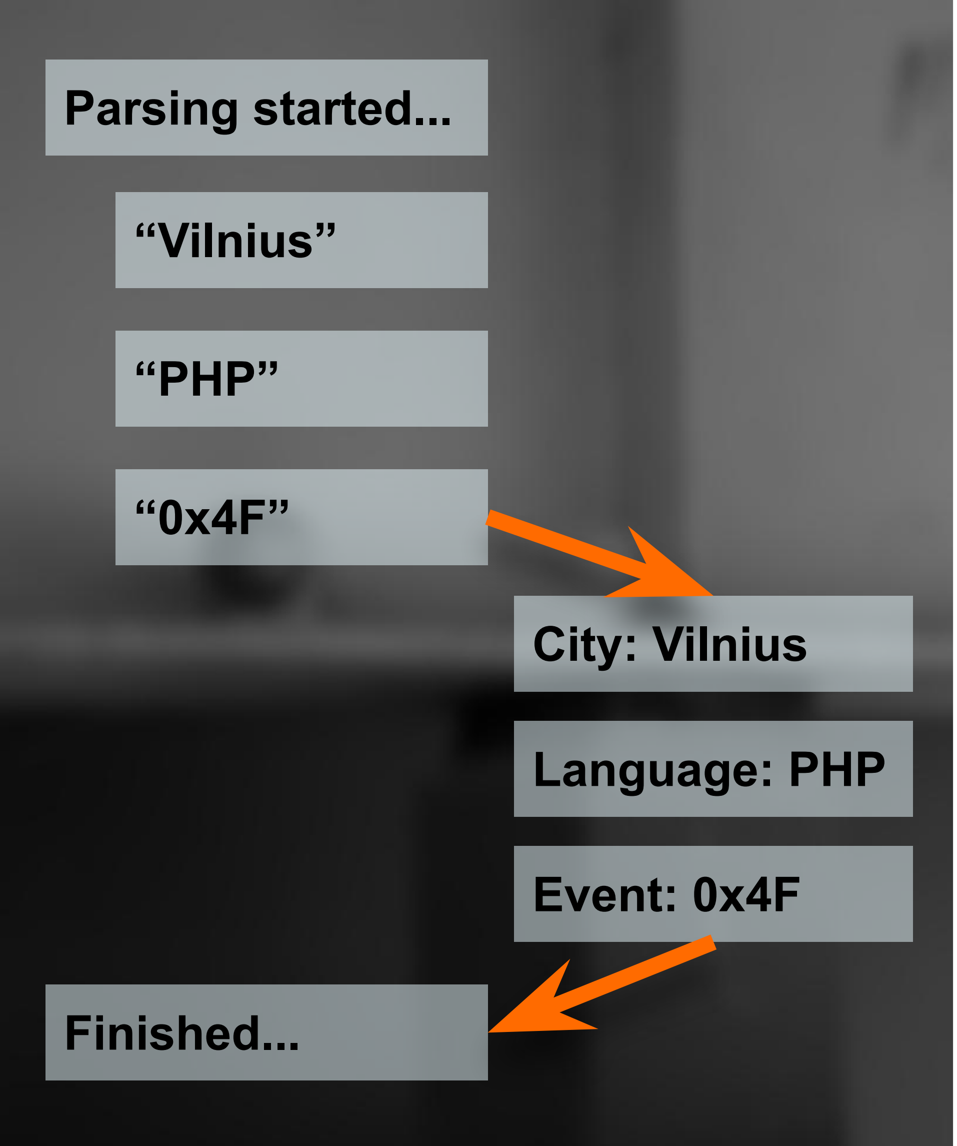
What are coroutines

HTTP

Why are coroutines useful

Production

How to use them in production



Parsing started...

“Vilnius”

“PHP”

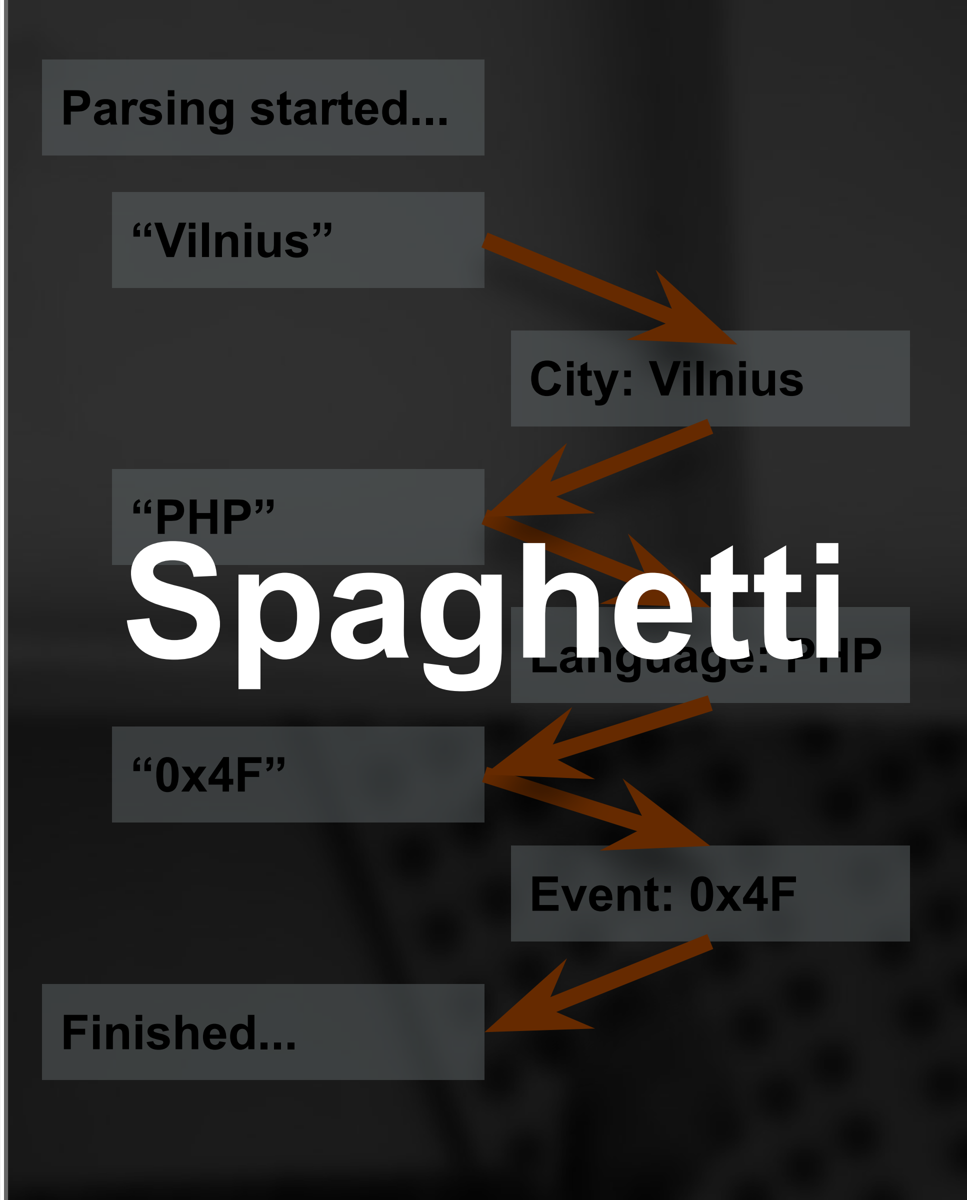
“0x4F”

City: Vilnius

Language: PHP

Event: 0x4F

Finished...



Parsing started...

“Vilnius”

City: Vilnius

“PHP”

Language: PHP

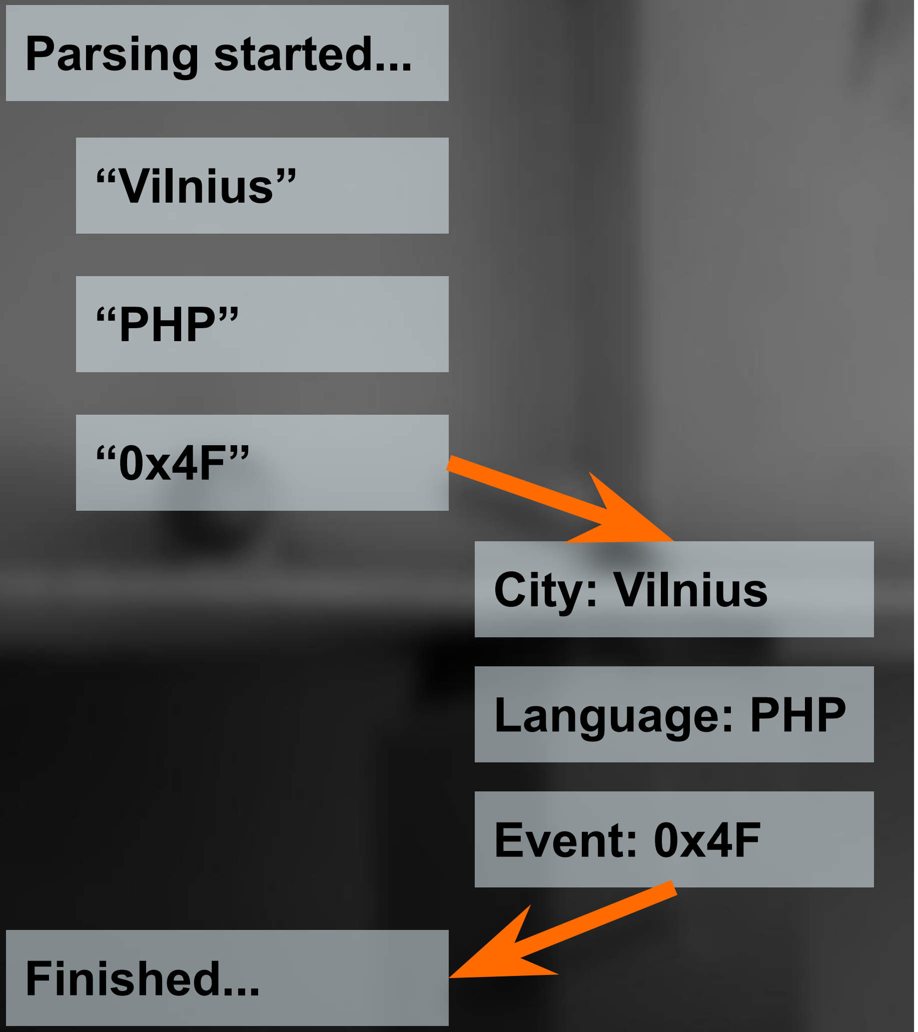
“0x4F”

Event: 0x4F

Finished...

Spaghetti

https://gist.github.com/aurelijusb/cacef3e4d57ca6e1772104e9a85079a9



Parsing started...

“Vilnius”

“PHP”

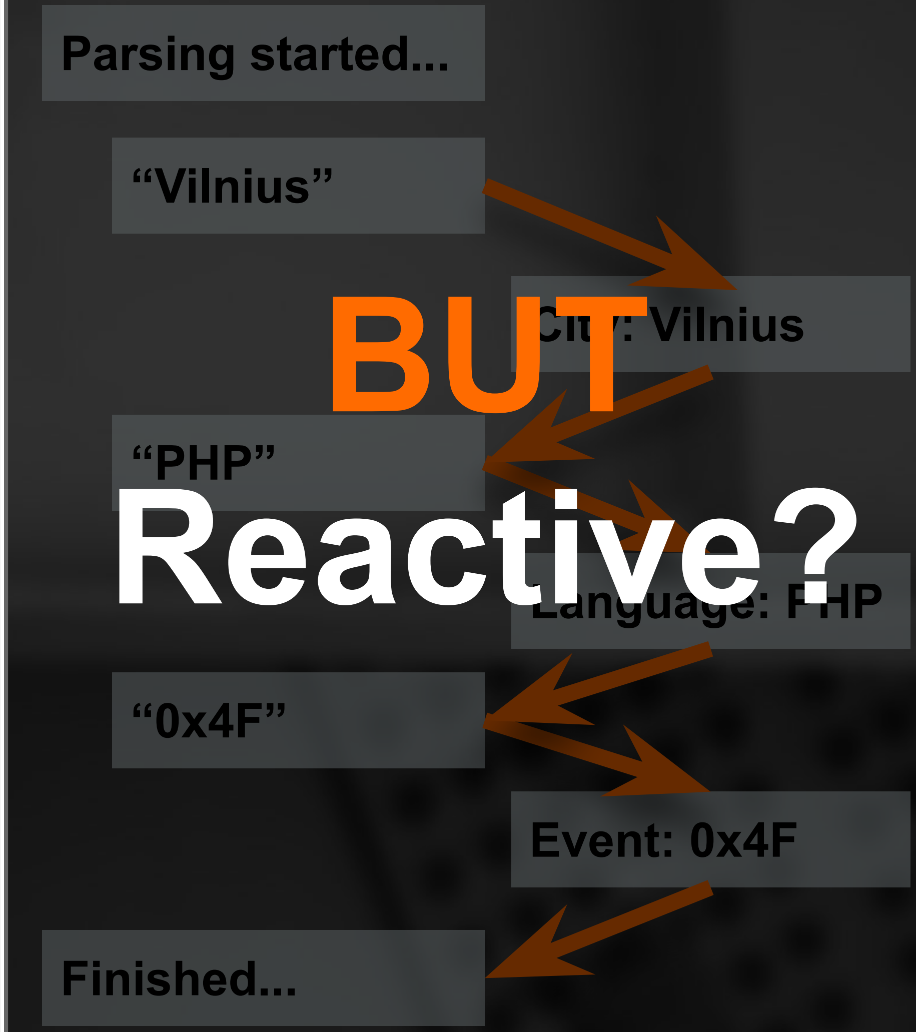
“0x4F”

City: Vilnius

Language: PHP

Event: 0x4F

Finished...



Parsing started...

“Vilnius”

City: Vilnius

“PHP”

Language: PHP

“0x4F”

Event: 0x4F

Finished...

BUT Reactive?

https://gist.github.com/aurelijusb/cacef3e4d57ca6e1772104e9a85079a9

Parse full

```
2019-06-05 17:35:57.484100 Receiving...
2019-06-05 17:36:02.490600 Received
2019-06-05 17:36:02.491000 Closed
2019-06-05 17:36:02.491200  Transformation started
2019-06-05 17:36:02.491300    Result: data: 0
2019-06-05 17:36:02.991700  Transformation finished
2019-06-05 17:36:02.991900  Transformation started
2019-06-05 17:36:02.992200    Result: data: 1
2019-06-05 17:36:03.492400  Transformation finished
2019-06-05 17:36:03.492600  Transformation started
2019-06-05 17:36:03.492700    Result: data: 2
2019-06-05 17:36:03.992900  Transformation finished
2019-06-05 17:36:03.993200  Transformation started
2019-06-05 17:36:03.993300    Result: data: 3
2019-06-05 17:36:04.493500  Transformation finished
2019-06-05 17:36:04.493800  Transformation started
2019-06-05 17:36:04.493900    Result: data: 4
2019-06-05 17:36:04.994200  Transformation finished
2019-06-05 17:36:04.994400  Transformation started
2019-06-05 17:36:04.994500    Result:
2019-06-05 17:36:05.494800  Transformation finished
2019-06-05 17:36:05.494900 Took: 8.01
```

https://gist.github.com/aurelijusb/cc18fc9e856b42753075906e3f96bffc

Parse chunked

```
2019-06-05 17:35:58.445000 Receiving...
2019-06-05 17:35:59.639100  <<< RECEIVED
2019-06-05 17:35:59.639200  Transformation started
2019-06-05 17:35:59.639300    Result: data: 0
2019-06-05 17:36:00.139500  Transformation finished
2019-06-05 17:36:00.639200  <<< RECEIVED
2019-06-05 17:36:00.639300  Transformation started
2019-06-05 17:36:00.639400    Result: data: 1
2019-06-05 17:36:01.139500  Transformation finished
2019-06-05 17:36:01.639600  <<< RECEIVED
2019-06-05 17:36:01.639800  Transformation started
2019-06-05 17:36:01.639900    Result: data: 2
2019-06-05 17:36:02.140100  Transformation finished
2019-06-05 17:36:02.639800  <<< RECEIVED
2019-06-05 17:36:02.640000  Transformation started
2019-06-05 17:36:02.640000    Result: data: 3
2019-06-05 17:36:03.140200  Transformation finished
2019-06-05 17:36:03.640200  <<< RECEIVED
2019-06-05 17:36:03.640400  Transformation started
2019-06-05 17:36:03.640600    Result: data: 4
2019-06-05 17:36:04.140800  Transformation finished
2019-06-05 17:36:04.141200 Received
2019-06-05 17:36:04.141400 Closed
2019-06-05 17:36:04.141500 Took: 5.7
```

```
curl_setopt($ch,
CURLOPT_RETURNTRANSFER, 1);

// ...

foreach ($result as $item) {
  transform($item);
}
```

```
2019-06-05 17:35:57.484100 Receiving...
2019-06-05 17:36:02.490600 Received
2019-06-05 17:36:02.491000 Closed
2019-06-05 17:36:02.491200  Transformation started
2019-06-05 17:36:02.491300    Result: data: 0
2019-06-05 17:36:02.991700  Transformation finished
2019-06-05 17:36:02.991900  Transformation started
2019-06-05 17:36:02.992200    Result: data: 1
2019-06-05 17:36:03.492400  Transformation finished
2019-06-05 17:36:03.492600  Transformation started
2019-06-05 17:36:03.492700    Result: data: 2
2019-06-05 17:36:03.992900  Transformation finished
2019-06-05 17:36:03.993200  Transformation started
2019-06-05 17:36:03.993300    Result: data: 3
2019-06-05 17:36:04.493500  Transformation finished
2019-06-05 17:36:04.493800  Transformation started
2019-06-05 17:36:04.493900    Result: data: 4
2019-06-05 17:36:04.994200  Transformation finished
2019-06-05 17:36:04.994400  Transformation started
2019-06-05 17:36:04.994500    Result:
2019-06-05 17:36:05.494800  Transformation finished
2019-06-05 17:36:05.494900 Took: 8.01
```

Link to Gist

https://gist.github.com/aurelijusb/cc18fc9e856b42753075906e3f96bffc

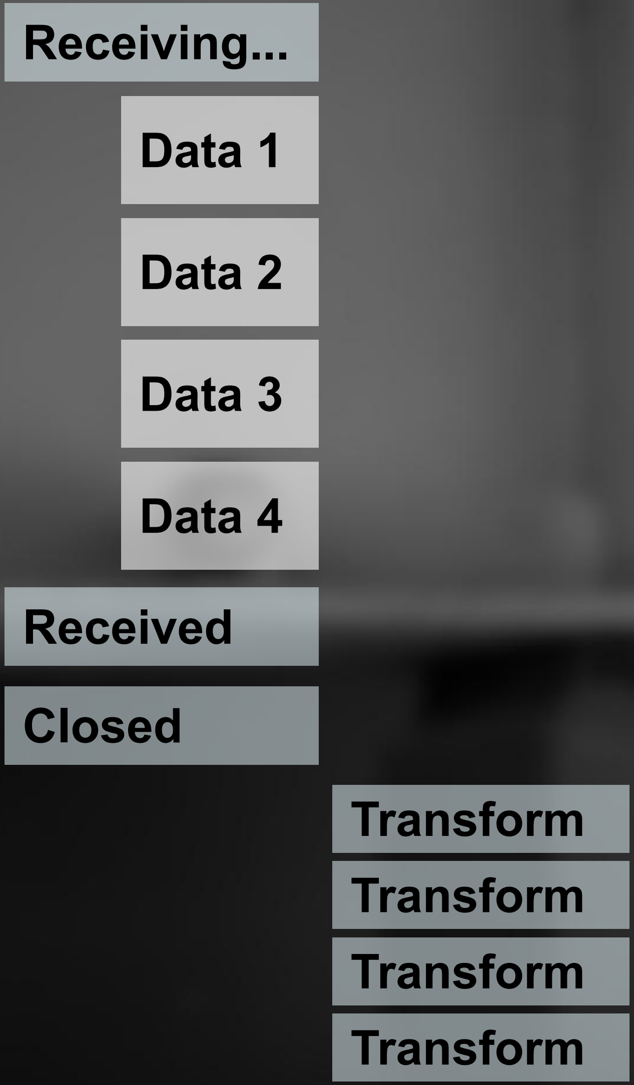
```
curl_setopt($ch,
CURLOPT_WRITEFUNCTION, $reader);

// ...

$reader = function ($curl, $data) {
  transform($data);
  return strlen($data);
};
```

Callable

```
2019-06-05 17:35:58.445000 Receiving...
2019-06-05 17:35:59.639100  <<< RECEIVED
2019-06-05 17:35:59.639200  Transformation started
2019-06-05 17:35:59.639300    Result: data: 0
2019-06-05 17:36:00.139500  Transformation finished
2019-06-05 17:36:00.639200  <<< RECEIVED
2019-06-05 17:36:00.639300  Transformation started
2019-06-05 17:36:00.639400    Result: data: 1
2019-06-05 17:36:01.139500  Transformation finished
2019-06-05 17:36:01.639600  <<< RECEIVED
2019-06-05 17:36:01.639800  Transformation started
2019-06-05 17:36:01.639900    Result: data: 2
2019-06-05 17:36:02.140100  Transformation finished
2019-06-05 17:36:02.639800  <<< RECEIVED
2019-06-05 17:36:02.640000  Transformation started
2019-06-05 17:36:02.640000    Result: data: 3
2019-06-05 17:36:03.140200  Transformation finished
2019-06-05 17:36:03.640200  <<< RECEIVED
2019-06-05 17:36:03.640400  Transformation started
2019-06-05 17:36:03.640600    Result: data: 4
2019-06-05 17:36:04.140800  Transformation finished
2019-06-05 17:36:04.141200 Received
2019-06-05 17:36:04.141400 Closed
2019-06-05 17:36:04.141500 Took: 5.7
```



Receiving...

Data 1

Data 2

Data 3

Data 4

Received

Closed

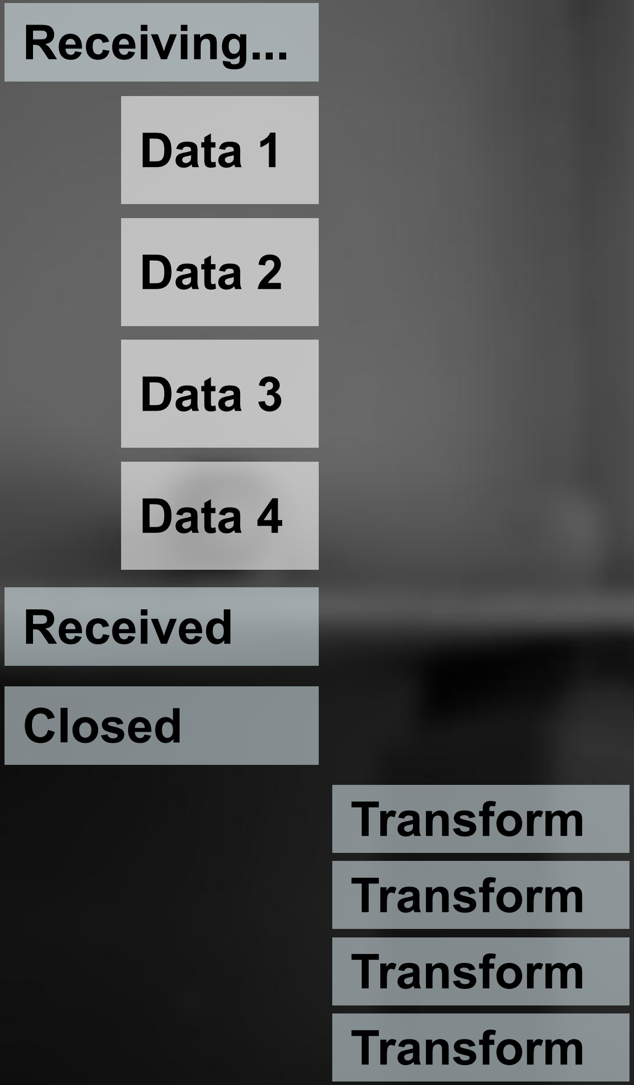
Transform

Transform

Transform

Transform

https://gist.github.com/aurelijusb/cc18fc9e856b42753075906e3f96bffc



Receiving...

Data 1

Data 2

Data 3

Data 4

Received

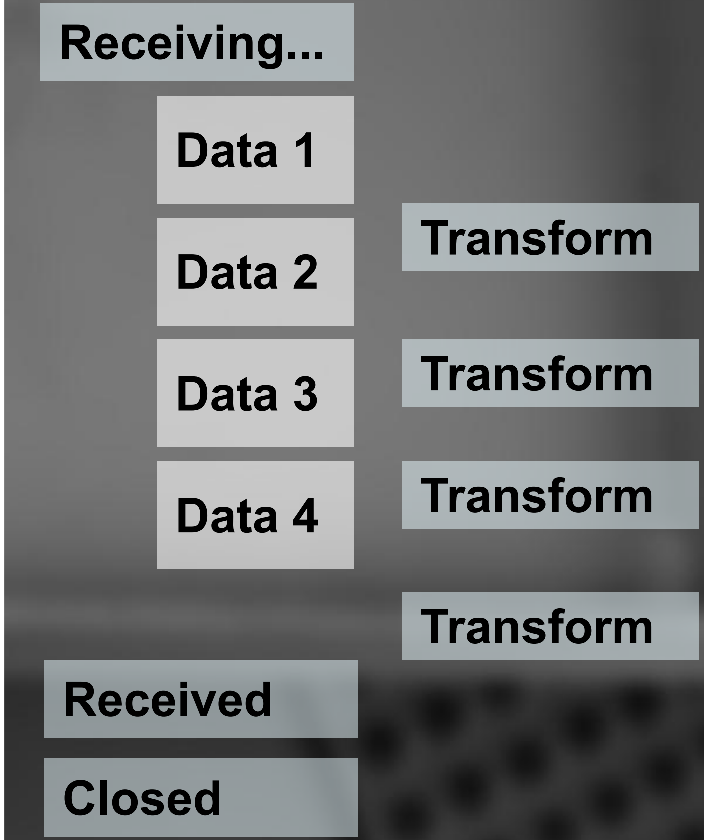
Closed

Transform

Transform

Transform

Transform



Receiving...

Data 1

Transform

Data 2

Transform

Data 3

Transform

Data 4

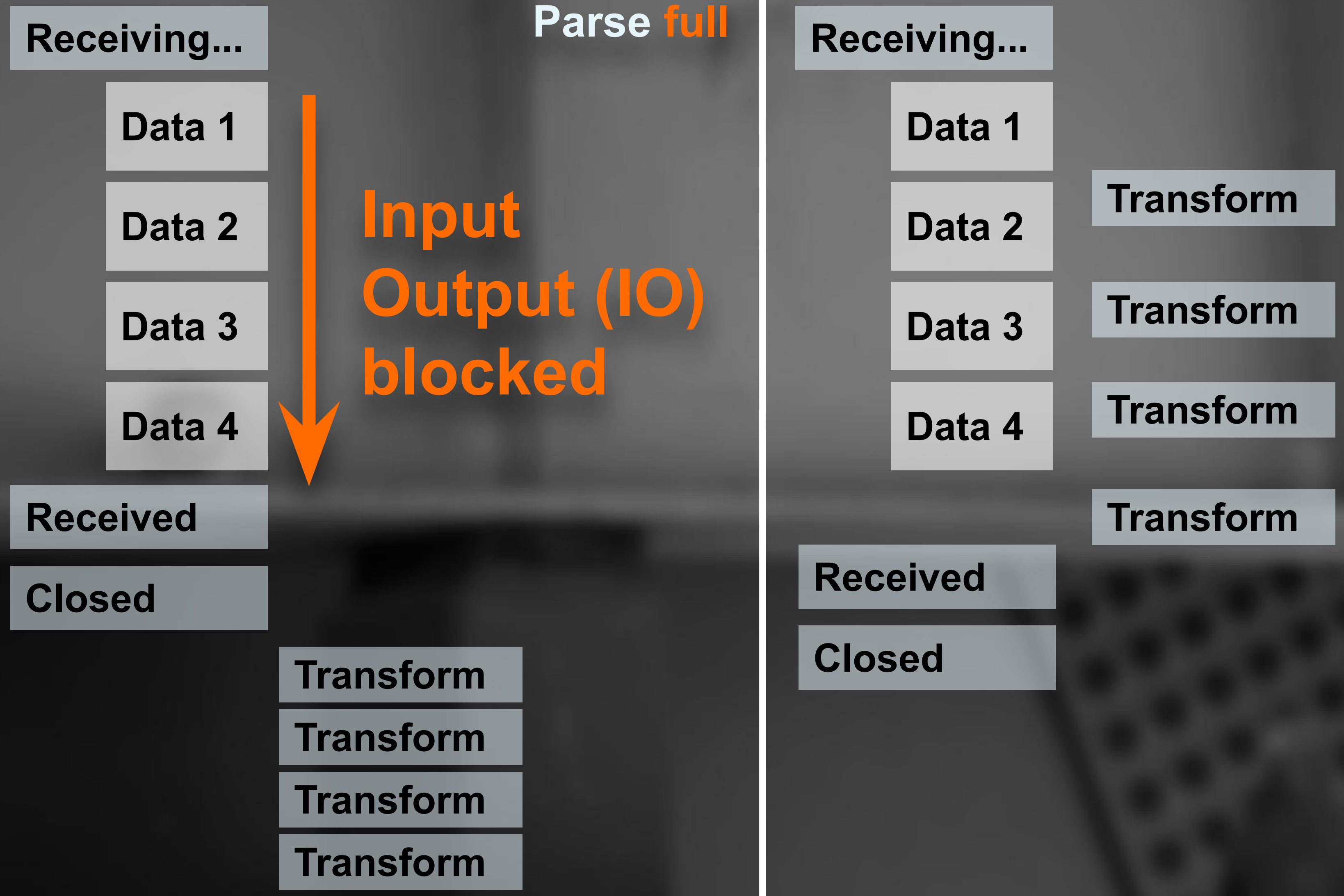
Transform

Received

Closed

https://gist.github.com/aurelijusb/cc18fc9e856b42753075906e3f96bffc

Parse full



Receiving...

Data 1

Data 2

Data 3

Data 4

Received

Closed

Input Output (IO) blocked

Transform

Transform

Transform

Transform

Receiving...

Data 1

Transform

Data 2

Transform

Data 3

Transform

Data 4

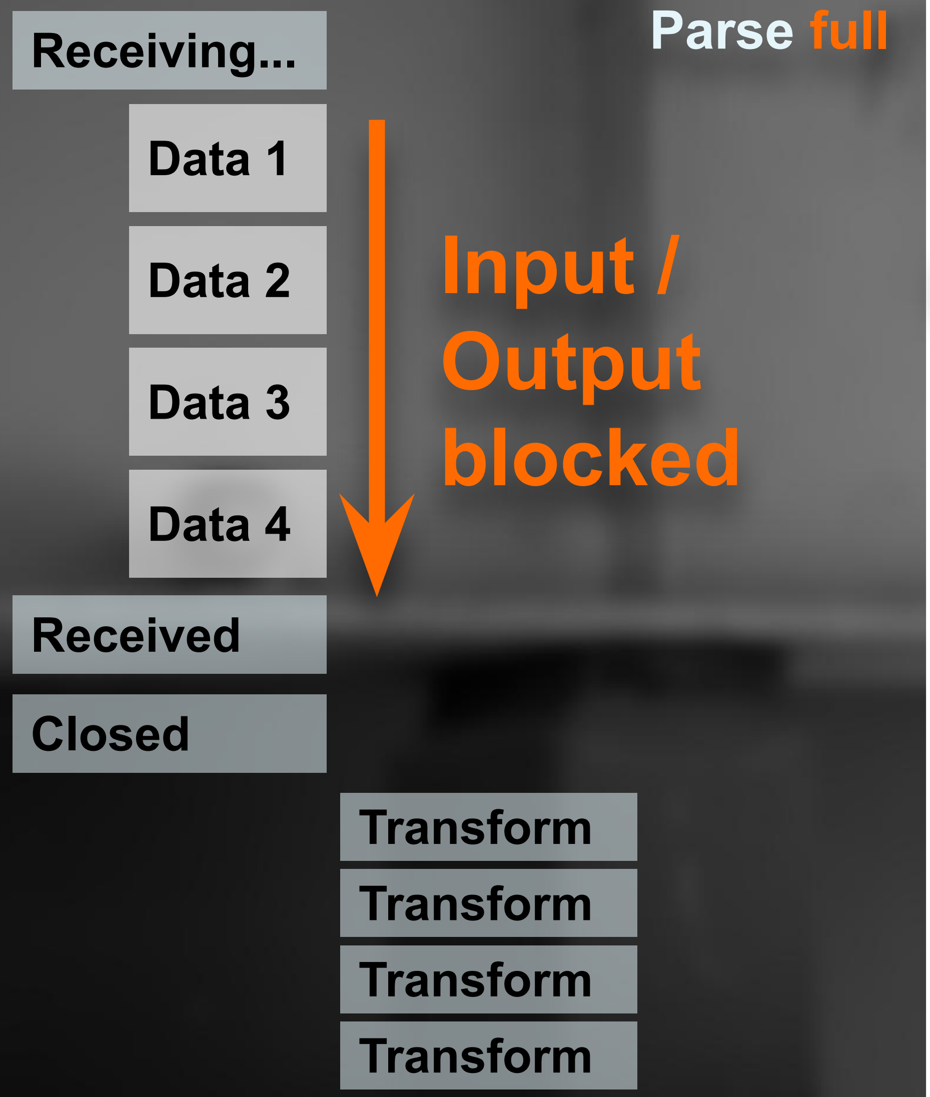
Transform

Received

Closed

https://gist.github.com/aurelijusb/cc18fc9e856b42753075906e3f96bffc

Parse full



Receiving...

Data 1

Data 2

Data 3

Data 4

Received

Closed

Input / Output blocked

Transform

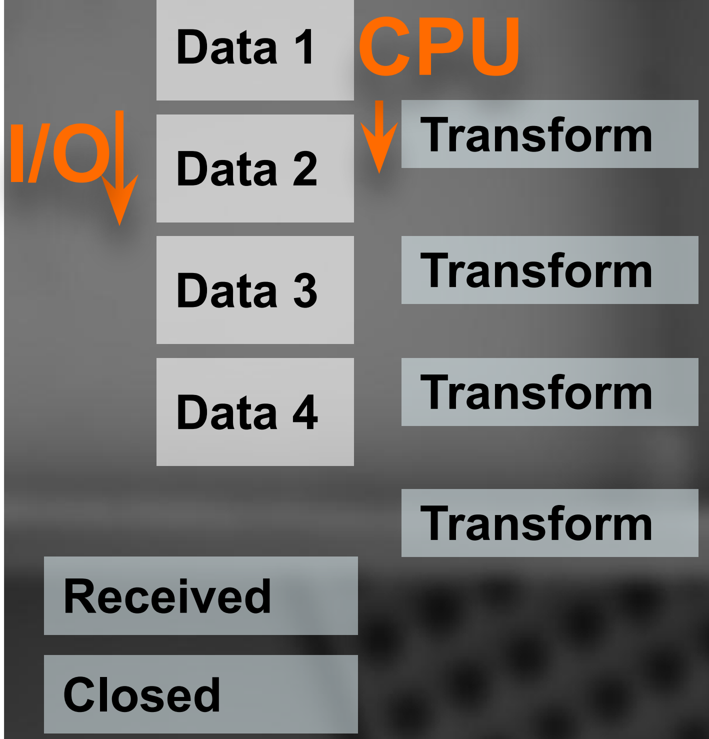
Transform

Transform

Transform

https://gist.github.com/aurelijusb/cc18fc9e856b42753075906e3f96bffc

Parse chunked



Receiving...

Data 1

CPU

Transform

I/O

Data 2

Data 3

Transform

Data 4

Transform

Transform

Received

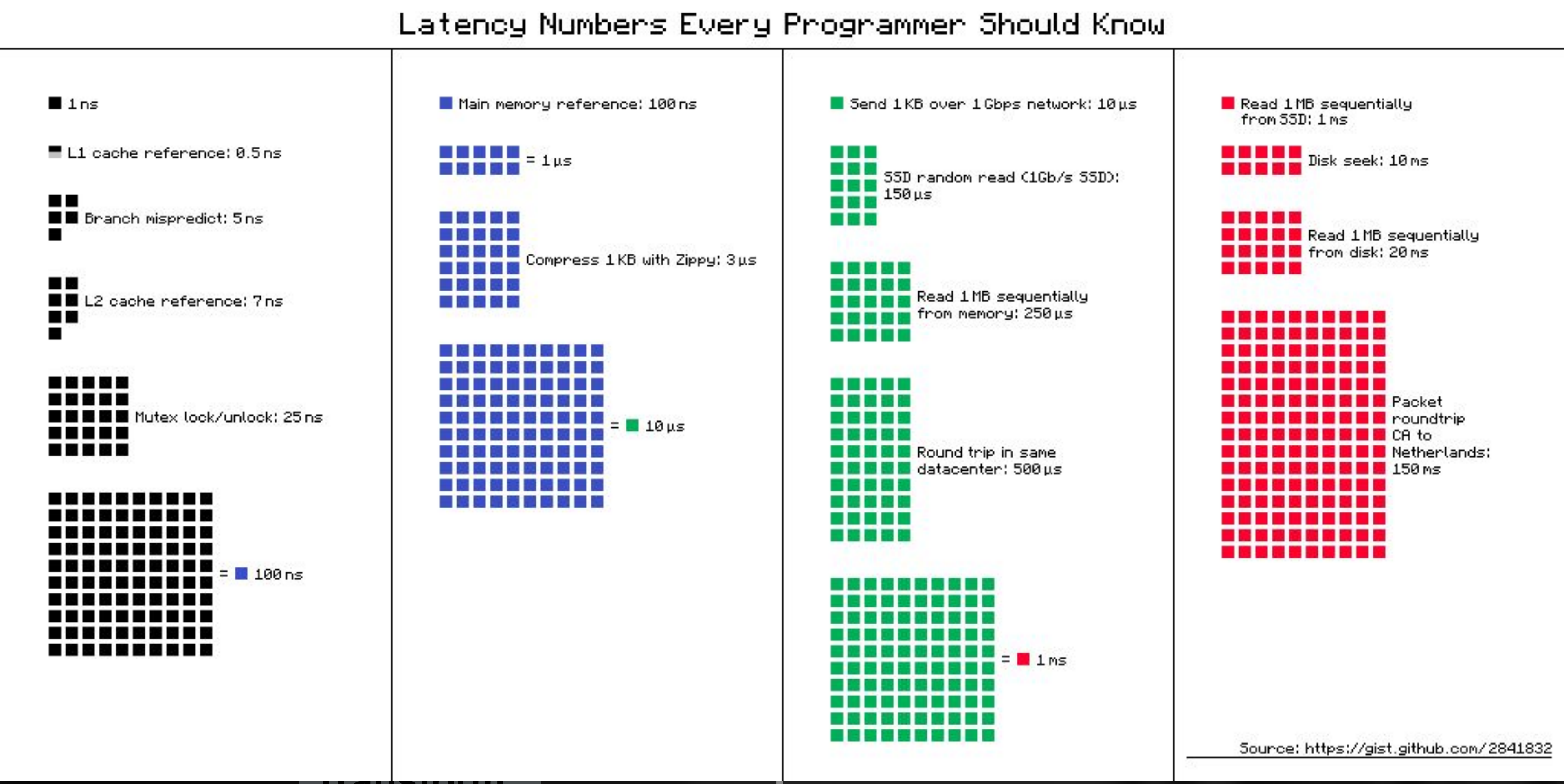
Closed

Receiving...

Parse **full**

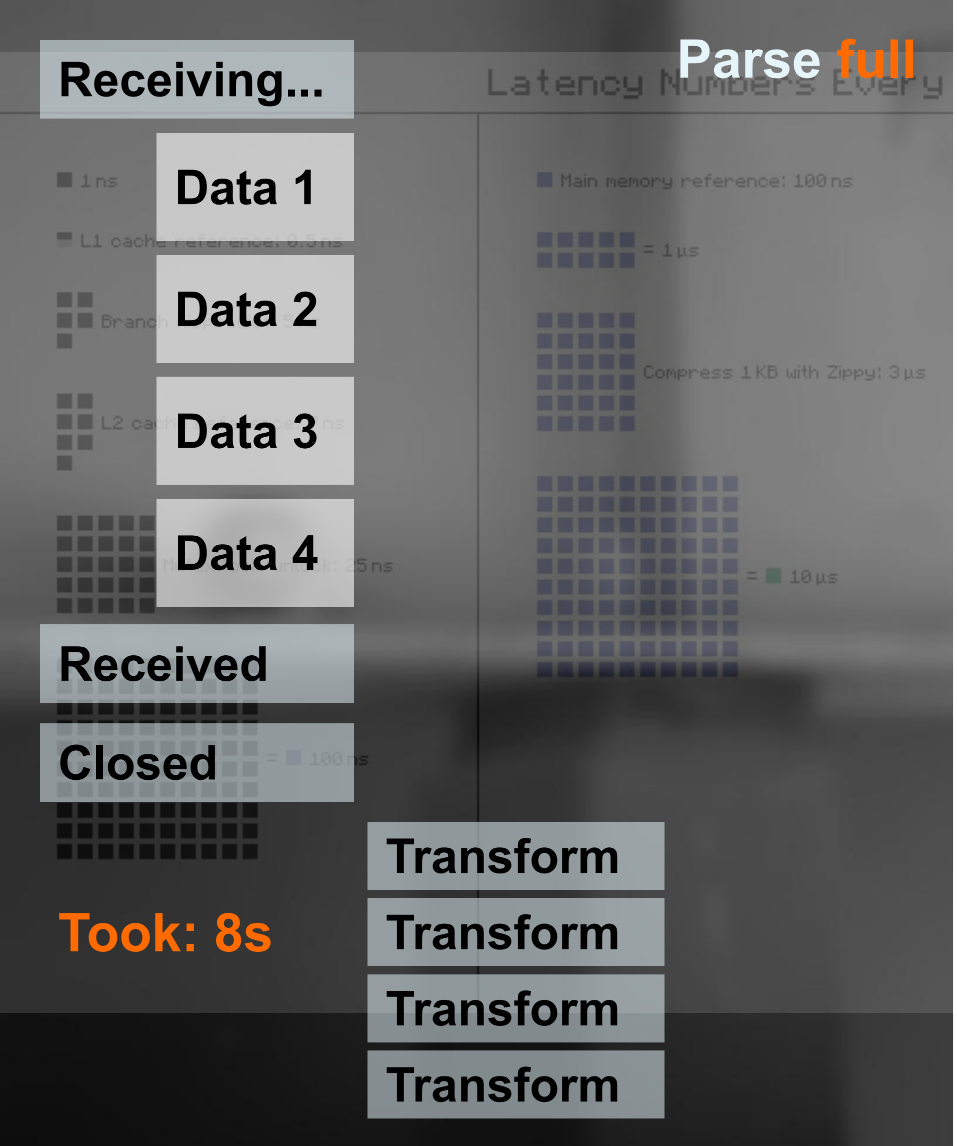
Receiving...

Parse **chunked**



Transform

https://gist.github.com/aurelijusb/cc18fc9e856b42753075906e3f96bffc



Parse **full**

Receiving...

Data 1

Data 2

Data 3

Data 4

Received

Closed

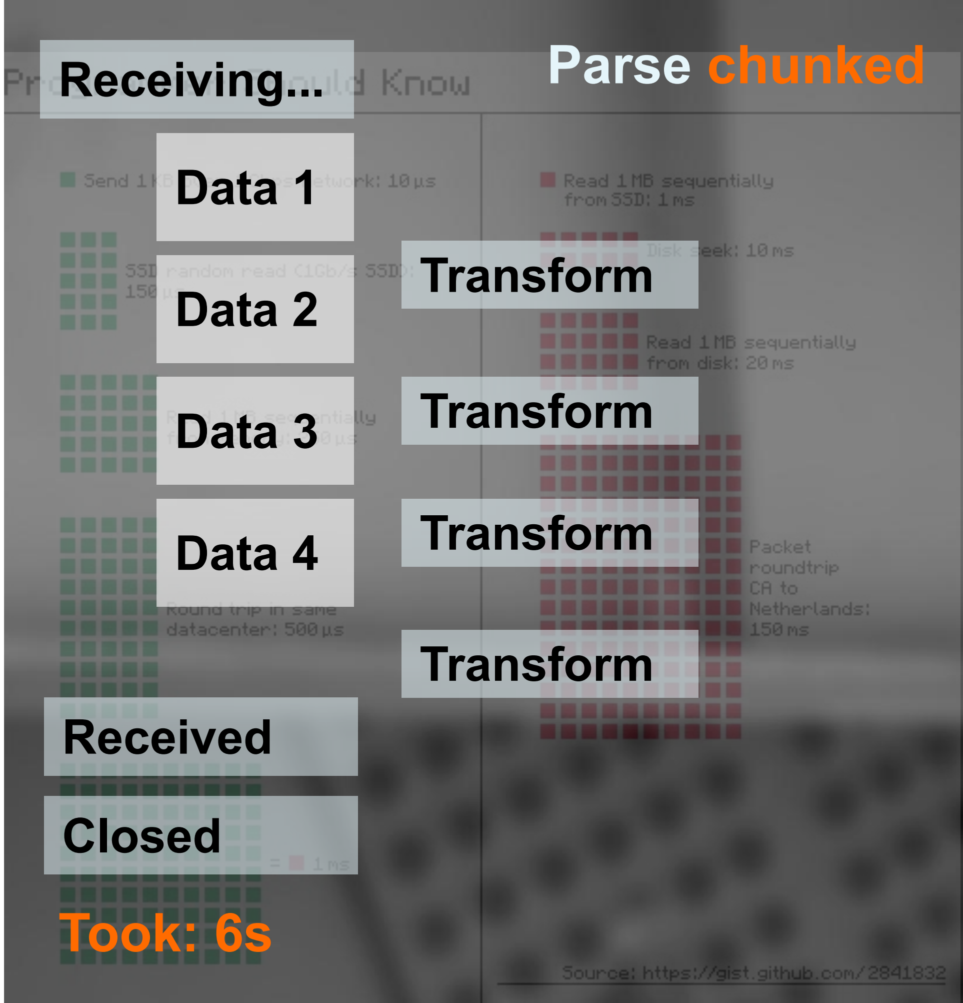
Transform

Transform

Transform

Transform

Took: 8s



Parse **chunked**

Receiving...

Data 1

Transform

Data 2

Transform

Data 3

Transform

Data 4

Transform

Received

Closed

Took: 6s

https://gist.github.com/aurelijusb/cc18fc9e856b42753075906e3f96bffc ←

Simple

What are coroutines

HTTP

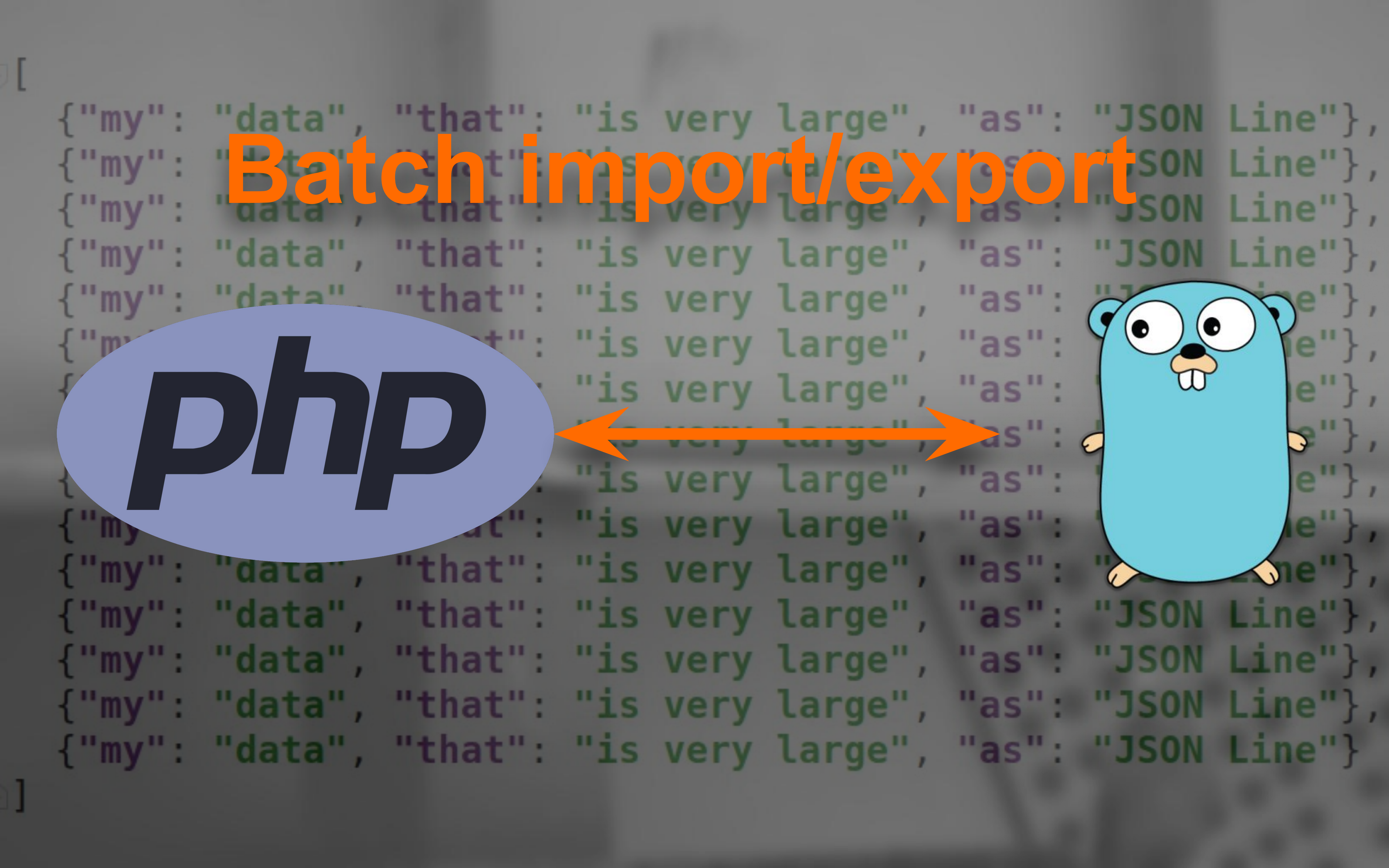
Async via chunks: Use CPU while I/O is blocking

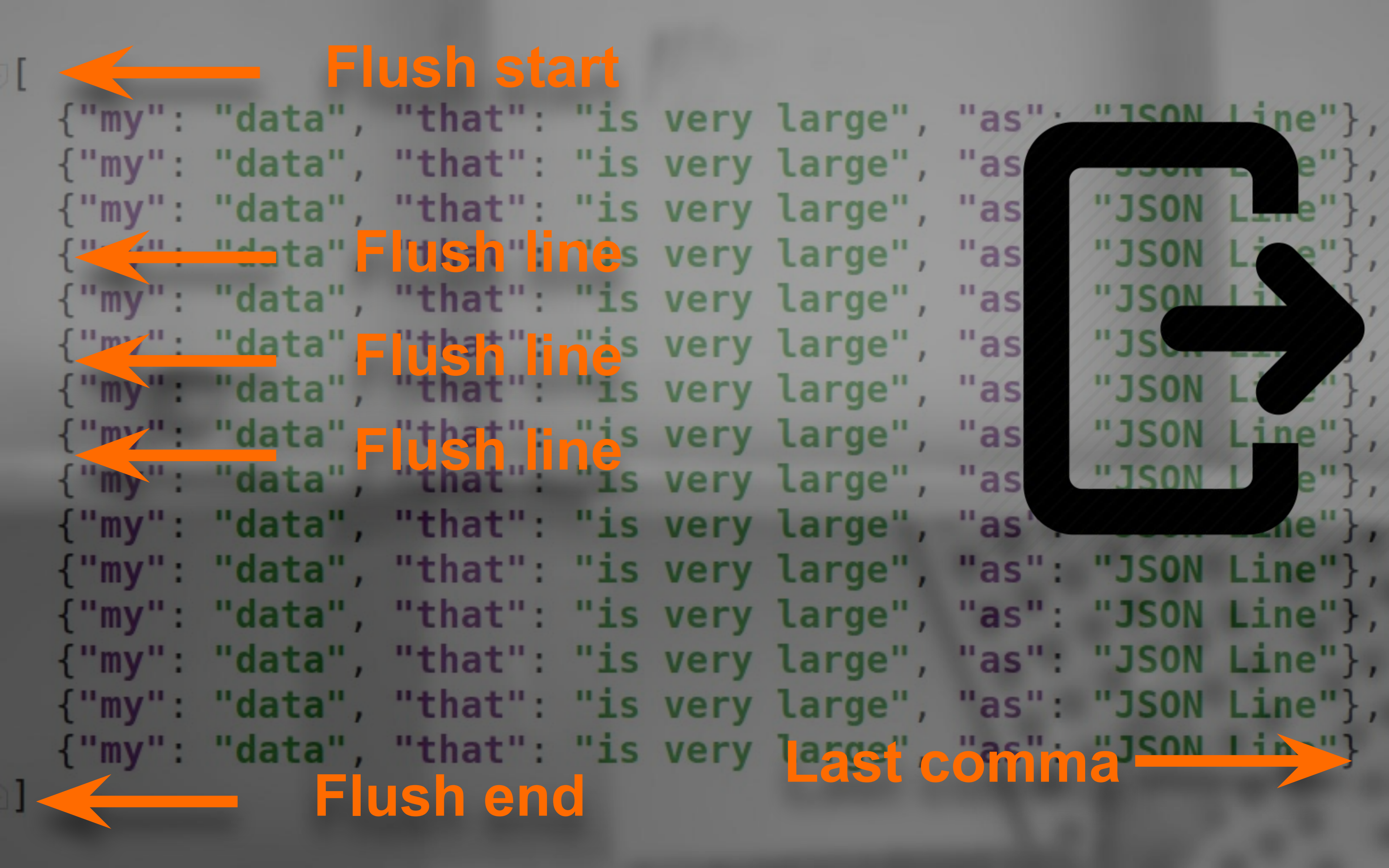
Production

How to use them in production

Simple	What are coroutines
HTTP	Why are coroutines useful
Production	How to use them in production

Batch import/export





```
[
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"}
]
```

Flush start

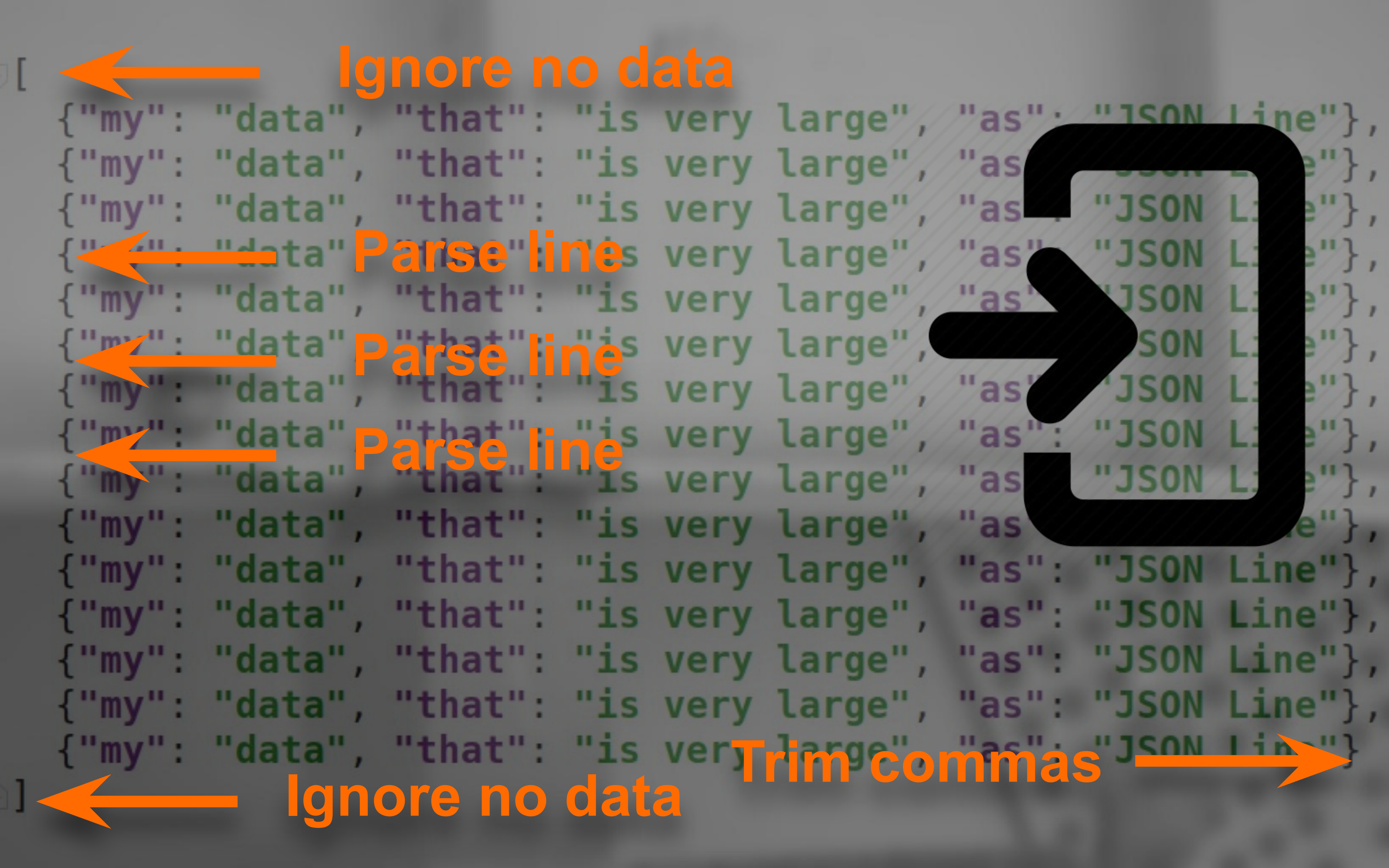
Flush line

Flush line

Flush line

Last comma

Flush end



```
[
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"}
]
```

Ignore no data

Parse line

Parse line

Parse line

Trim commas

Ignore no data

Simple	What are coroutines
HTTP	Why are coroutines useful
Production	Flush/read JSON items as lines

Async is complex

Rely on basics

```
[
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"},
  {"my": "data", "that": "is very large", "as": "JSON Line"}
]
```

Thank you

Questions?

Coroutines in PHP

Aurelijus Banelis

VilniusPHP 0x4F
2019-06-06

NFQ is hiring
Ping me – split bonus

References / further reading

- https://en.wikipedia.org/wiki/Coroutine
- https://wiki.php.net/rfc/generators
- https://gist.github.com/aurelijusb/cacef3e4d57ca6e1772104e9a85079a9
- https://gist.github.com/aurelijusb/cc18fc9e856b42753075906e3f96bffc
- https://symfony.com/doc/current/best_practices/tests.html#functional-tests
- https://amphp.org/