11.07.2024

# How I started contributing to php-src

And how everyone can as well

Adam Spychala
/jorgsowa

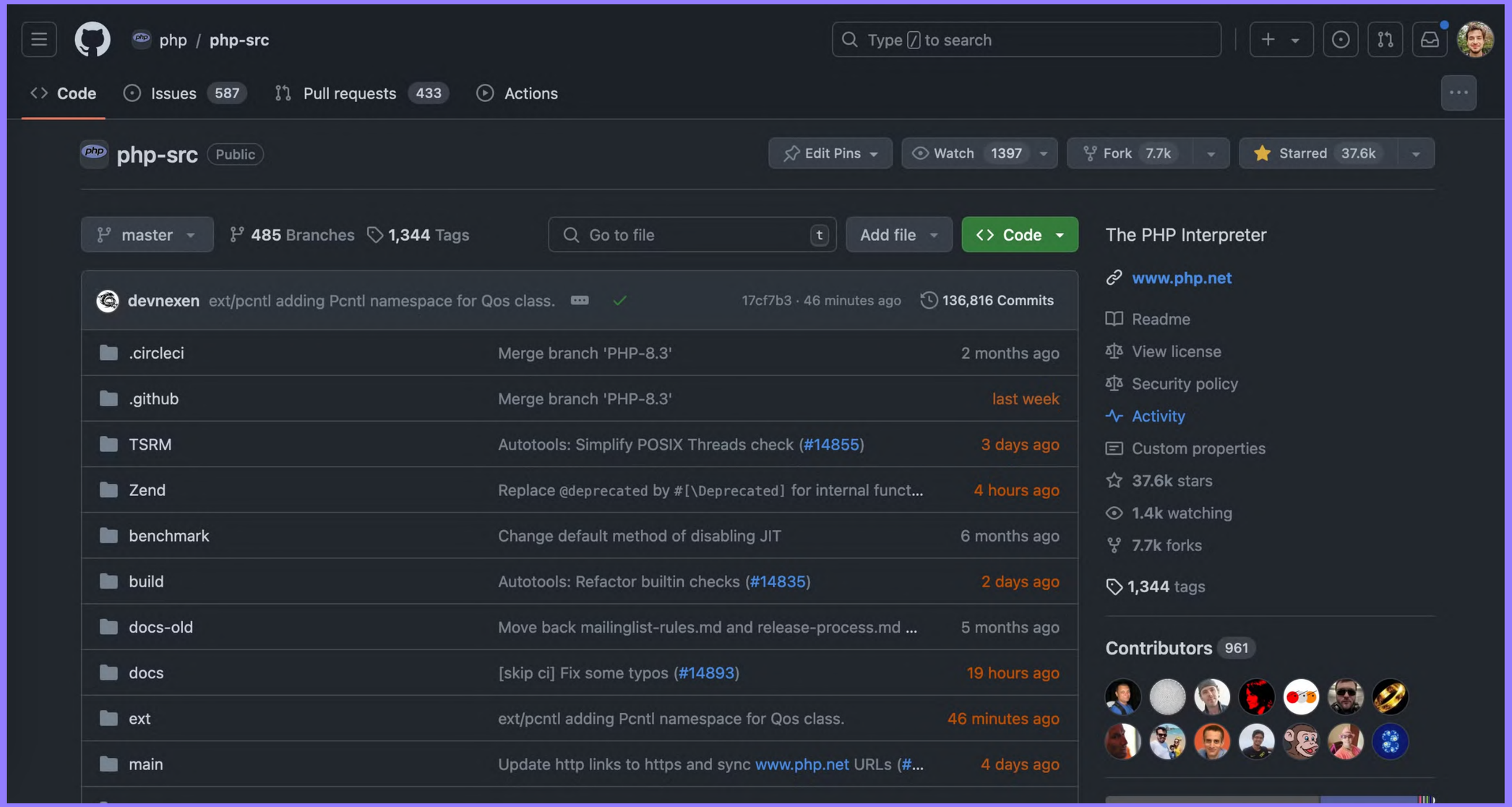
Try Pitch

# Where the repository of PHP code is located?

github.com/php/php-src

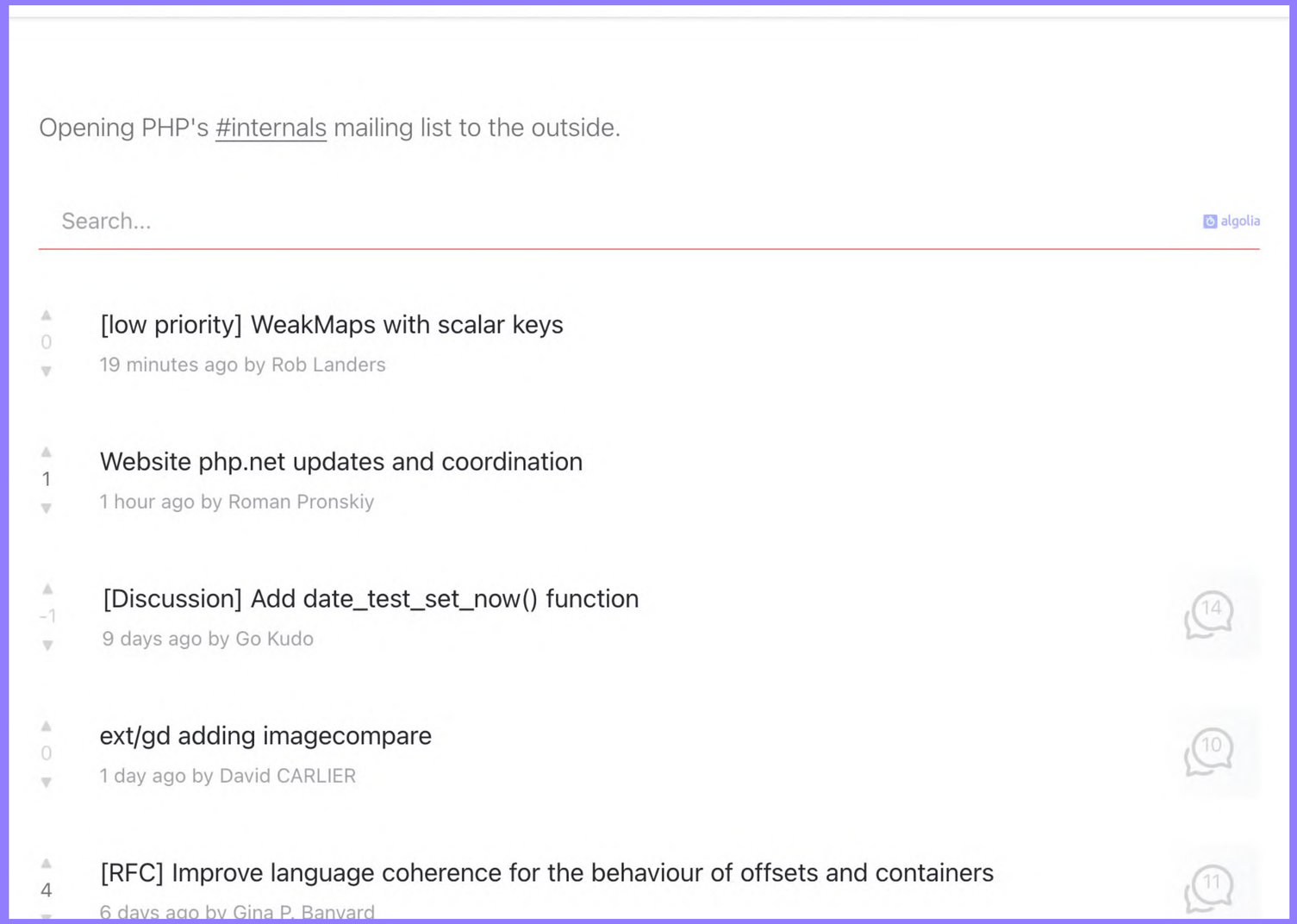
git.php.net

Try Pitch



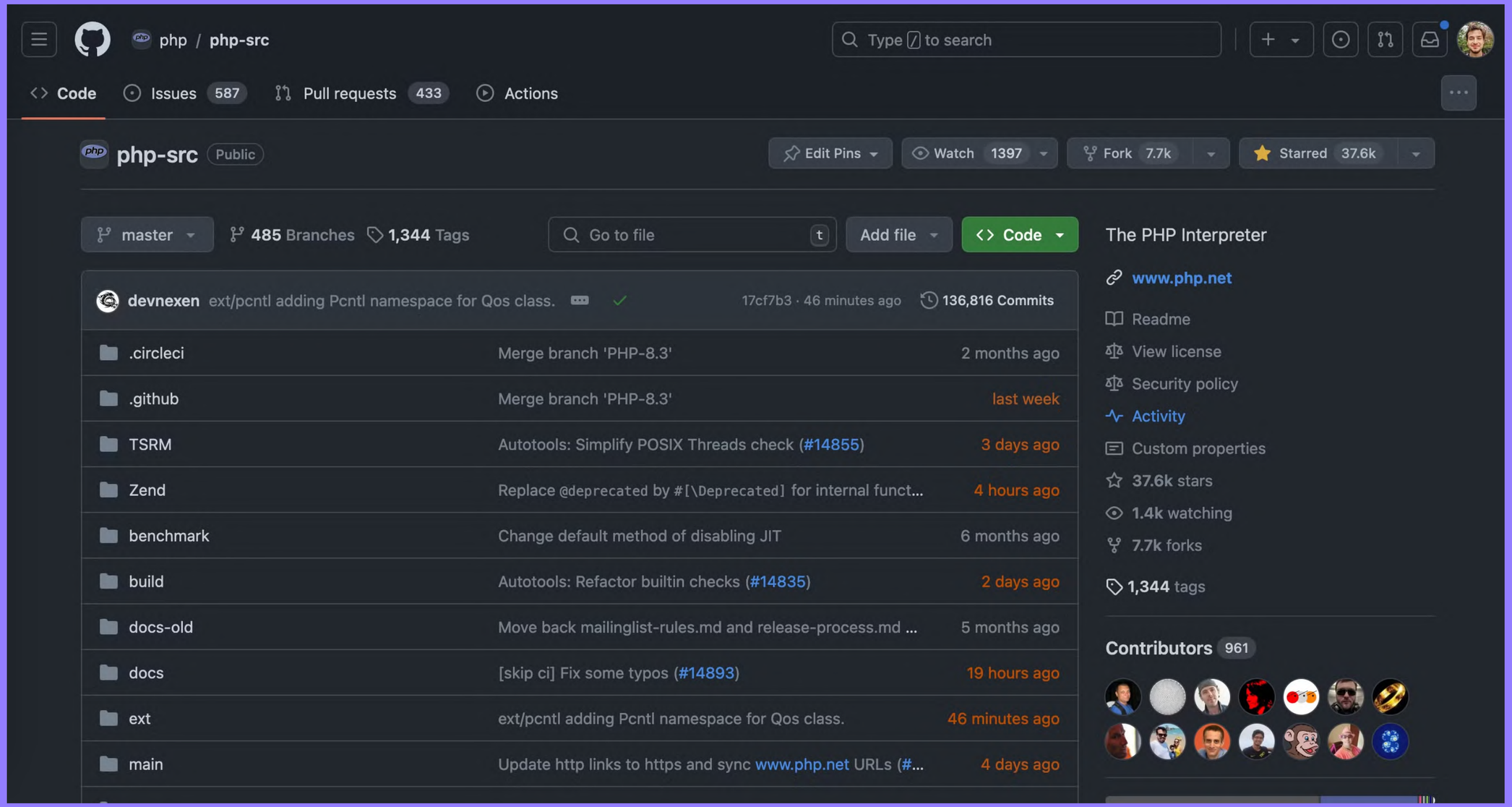

https://github.com/php/php-src

Try Pitch




https://externals.io/

Try Pitch




https://github.com/php/php-src

Try Pitch

## User Contributed Notes 32 notes

+ add a note

▲ 319 ▼ takingsides at gmail dot com — 10 years ago

In my opinion this function lacks two flags:

- PHP_ROUND_UP - Always round up.
- PHP_ROUND_DOWN - Always round down.

In accounting, it's often necessary to always round up, or down to a precision of thousandths.

```php
<?php
function round_up($number, $precision = 2)
{
    $fig = (int) str_pad('1', $precision, '0');
    return (ceil($number * $fig) / $fig);
}

function round_down($number, $precision = 2)
{
    $fig = (int) str_pad('1', $precision, '0');
    return (floor($number * $fig) / $fig);
}
?>
```

https://www.php.net/manual/en/function.round.php

Try Pitch

Change language: English

# round

(PHP 4, PHP 5, PHP 7, PHP 8)
round — Rounds a float

## Description

```
round(int|float $num, int $precision = 0, int $mode = PHP_ROUND_HALF_UP): float
```

Returns the rounded value of **num** to specified **precision** (number of digits after the decimal point). **precision** can also be negative or zero (default).

## Parameters

**num**
The value to round.

**precision**
The optional number of decimal digits to round to.

If the **precision** is positive, **num** is rounded to **precision** significant digits after the decimal point.

If the **precision** is negative, **num** is rounded to **precision** significant digits before the decimal point, i.e. to the nearest multiple of `pow(10, -$precision)`, e.g. for a **precision** of -1 **num** is rounded to tens, for a **precision** of -2 to hundreds, etc.

**mode**
Use one of the following constants to specify the mode in which rounding occurs.

| Constants | Description |
|---|---|
| PHP_ROUND_HALF_UP | Rounds **num** away from zero when it is half way there, making 1.5 into 2 and -1.5 into -2. |
| PHP_ROUND_HALF_DOWN | Rounds **num** towards zero when it is half way there, making 1.5 into 1 and -1.5 into -1. |
| PHP_ROUND_HALF_EVEN | Rounds **num** towards the nearest even value when it is half way there, making both 1.5 and 2.5 into 2. |
| PHP_ROUND_HALF_ODD | Rounds **num** towards the nearest odd value when it is half way there, making 1.5 into 1 and 2.5 into 3. |

## Return Values

The value rounded to the given **precision** as a float.

https://www.php.net/manual/en/function.round.php

Try Pitch

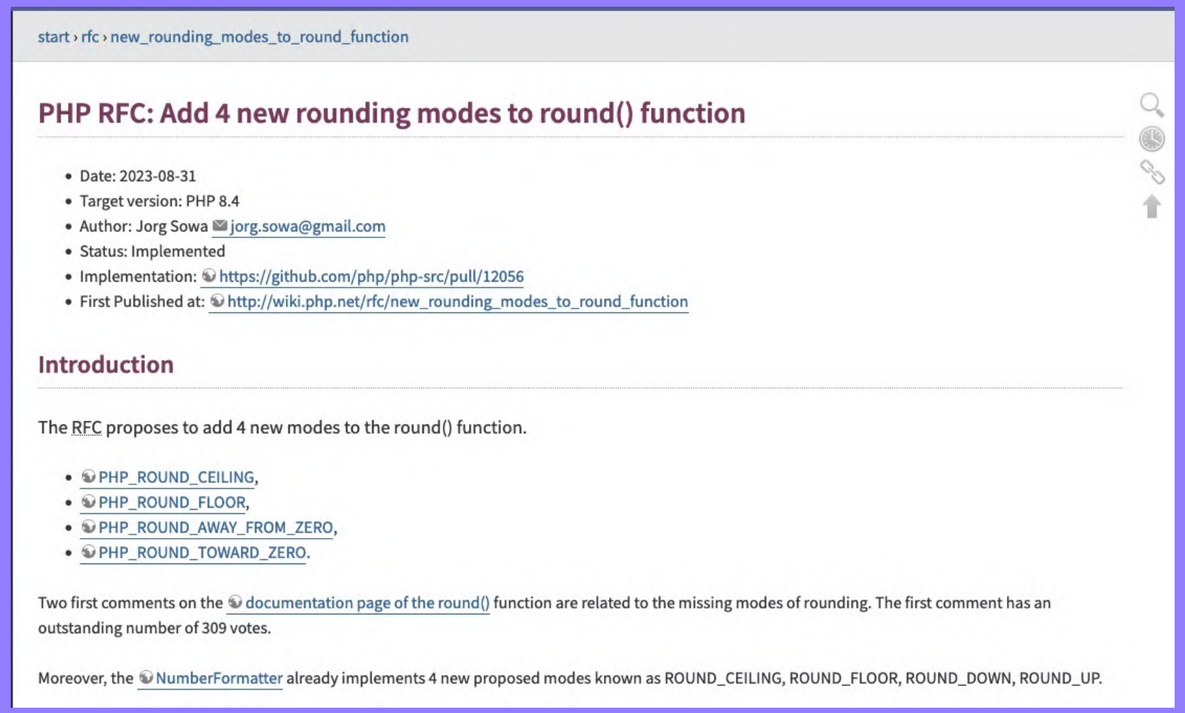
# Stages of adding new features to PHP

Try Pitch

Stage 0: Ask for Karma

Try Pitch

Stage 1: Write RFC



start › rfc › new_rounding_modes_to_round_function

# PHP RFC: Add 4 new rounding modes to round() function

- Date: 2023-08-31
- Target version: PHP 8.4
- Author: Jorg Sowa jorg.sowa@gmail.com
- Status: Implemented
- Implementation: https://github.com/php/php-src/pull/12056
- First Published at: http://wiki.php.net/rfc/new_rounding_modes_to_round_function

## Introduction

The RFC proposes to add 4 new modes to the round() function.

- PHP_ROUND_CEILING,
- PHP_ROUND_FLOOR,
- PHP_ROUND_AWAY_FROM_ZERO,
- PHP_ROUND_TOWARD_ZERO.

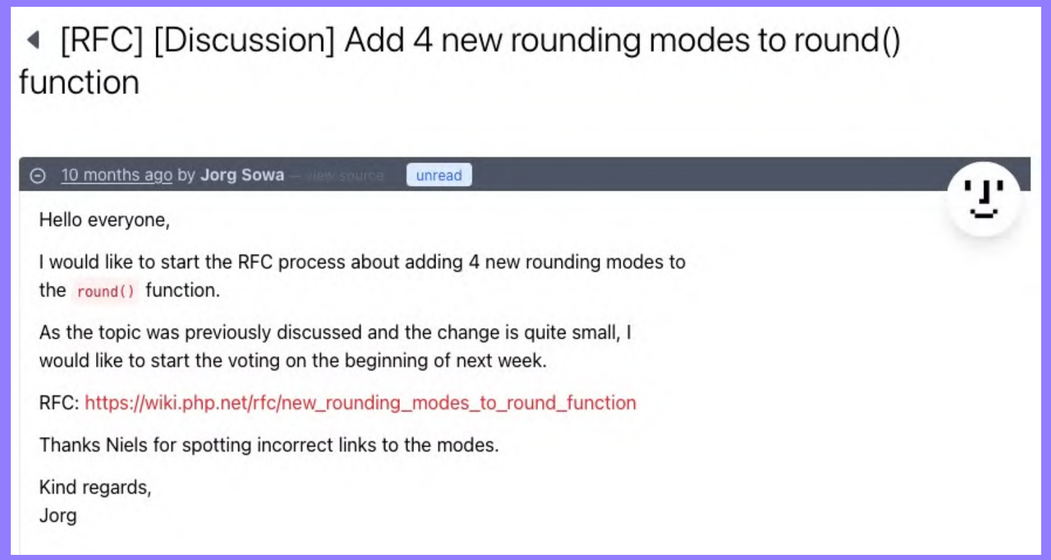
Two first comments on the documentation page of the round() function are related to the missing modes of rounding. The first comment has an outstanding number of 309 votes.

Moreover, the NumberFormatter already implements 4 new proposed modes known as ROUND_CEILING, ROUND_FLOOR, ROUND_DOWN, ROUND_UP.

https://wiki.php.net/rfc/new_rounding_modes_to_round_function

Try Pitch

Stage 2: Open discussion




Try Pitch

Stage 3: Discuss

1. Open your RFC for discussion.
2. Answer the questions.
3. Revisit your RFC upon appearing questions.
4. Modify your RFC.
5. Wait a few days for new questions.
6. If new questions appear, go to 2.
7. End the discussion stage.

Try Pitch

# Stage 4: Voting

As per the voting RFC a yes/no vote with a 2/3 majority is needed for this proposal to be accepted.

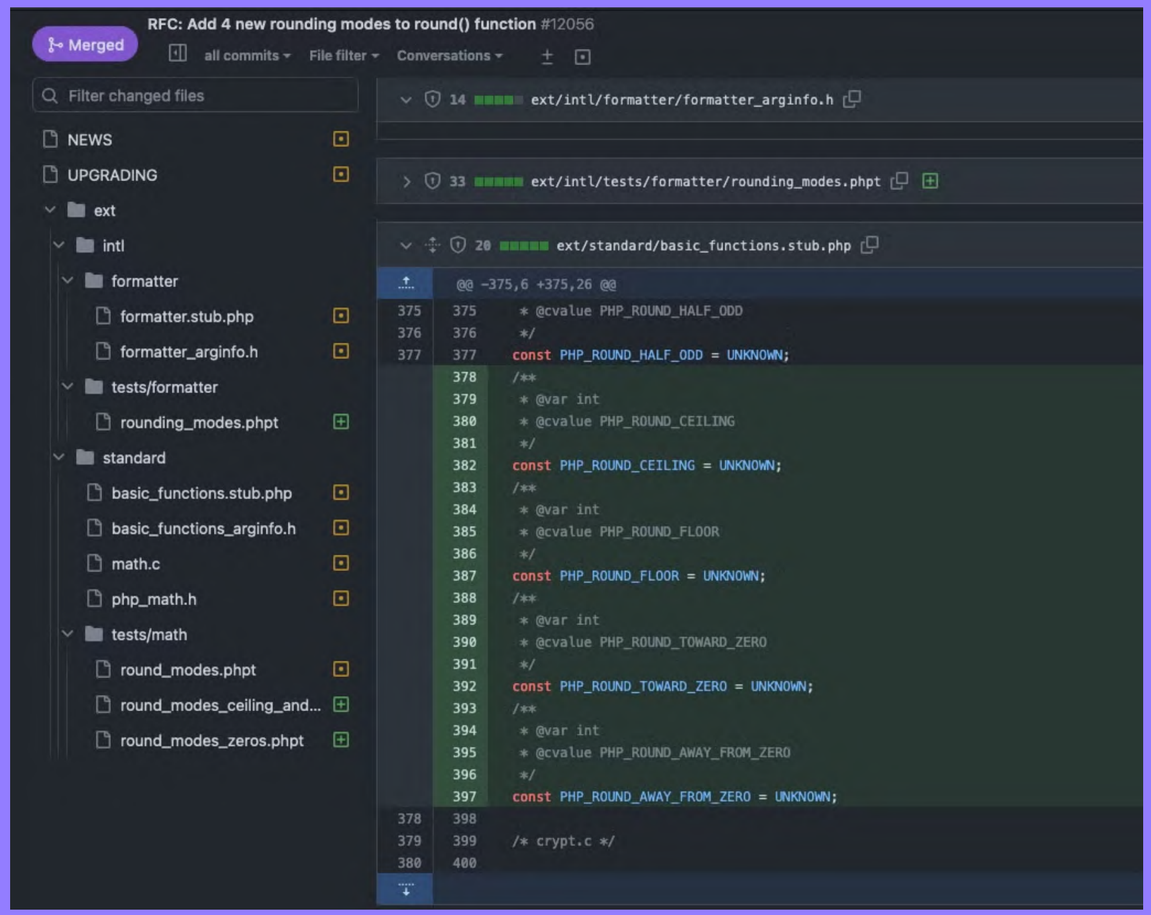
Voting started on 2023-11-15 and will end on 2023-11-30 00:00 GMT.

| Implement 4 new rounding modes to `round()` function | | |
|---|---|---|
| **Real name** | Yes | No |
| ashnazg (ashnazg) | ✓ | |
| bukka (bukka) | ✓ | |
| crell (crell) | ✓ | |
| derick (derick) | ✓ | |
| galvao (galvao) | ✓ | |
| girgias (girgias) | ✓ | |
| joey (joey) | ✓ | |
| kalle (kalle) | ✓ | |
| levim (levim) | ✓ | |
| mbeccati (mbeccati) | ✓ | |
| nicolasgrekas (nicolasgrekas) | ✓ | |
| nielsdos (nielsdos) | ✓ | |
| ocramius (ocramius) | ✓ | |
| petk (petk) | ✓ | |
| ramsey (ramsey) | ✓ | |
| santiagolizardo (santiagolizardo) | ✓ | |
| sergey (sergey) | ✓ | |
| theodorejb (theodorejb) | ✓ | |
| timwolla (timwolla) | ✓ | |
| **Final result:** | **19** | **0** |
| This poll has been closed. | | |

https://wiki.php.net/rfc/new_rounding_modes_to_round_function

Try Pitch

Stage 5: Implementation ...or acceptance of the results




https://github.com/php/php-src/pull/12056

Try Pitch

Implementation

```
315  341  /* {{{ Returns the number rounded to specified precision */
316  342  PHP_FUNCTION(round)
317  343  {
318  344      zval *value;
319  345      int places = 0;
320  346      zend_long precision = 0;
321  347      zend_long mode = PHP_ROUND_HALF_UP;
322  348
323  349      ZEND_PARSE_PARAMETERS_START(1, 3)
324  350          Z_PARAM_NUMBER(value)
325  351          Z_PARAM_OPTIONAL
326  352          Z_PARAM_LONG(precision)
327  353          Z_PARAM_LONG(mode)
328  354      ZEND_PARSE_PARAMETERS_END();
329  355
330  356      if (ZEND_NUM_ARGS() >= 2) {
331  357          if (precision >= 0) {
332  358              places = ZEND_LONG_INT_OVFL(precision) ? INT_MAX : (int)precision;
333  359          } else {
334  360              places = ZEND_LONG_INT_UDFL(precision) ? INT_MIN : (int)precision;
335  361          }
336  362      }
337  363
338  364      switch (mode) {
339  365          case PHP_ROUND_HALF_UP:
340  366          case PHP_ROUND_HALF_DOWN:
341  367          case PHP_ROUND_HALF_EVEN:
342  368          case PHP_ROUND_HALF_ODD:
     369          case PHP_ROUND_AWAY_FROM_ZERO:
     370          case PHP_ROUND_TOWARD_ZERO:
     371          case PHP_ROUND_CEILING:
     372          case PHP_ROUND_FLOOR:
343  373              break;
344  374          default:
345  375              zend_argument_value_error(3, "must be a valid rounding mode (PHP_ROUND_*)");
346  376              RETURN_THROWS();
347  377      }
348  378
349  379      switch (Z_TYPE_P(value)) {
350  380          case IS_LONG:
351  381              /* Simple case - long that doesn't need to be rounded. */
352  382              if (places >= 0) {
353  383                  RETURN_DOUBLE(zval_get_double(value));
354  384              }
355  385              ZEND_FALLTHROUGH;
356  386
357  387          case IS_DOUBLE:
358  388              RETURN_DOUBLE(_php_math_round(zval_get_double(value), (int)places, (int)mode));
359  389
360  390          EMPTY_SWITCH_DEFAULT_CASE();
361  391      }
362  392  }
363  393  /* }}} */
```

Try Pitch

Implementation

30 ext/standard/math.c

```
126 126
127 127             return integral;
128 128
    129         case PHP_ROUND_CEILING:
    130             if (value > 0.0 && fractional > 0.0) {
    131                 return integral + 1.0;
    132             }
    133
    134             return integral;
    135
    136         case PHP_ROUND_FLOOR:
    137             if (value < 0.0 && fractional > 0.0) {
    138                 return integral - 1.0;
    139             }
    140
    141             return integral;
    142
    143         case PHP_ROUND_TOWARD_ZERO:
    144             return integral;
    145
    146         case PHP_ROUND_AWAY_FROM_ZERO:
    147             if (fractional > 0.0) {
    148                 return integral + copysign(1.0, integral);
    149             }
    150
    151             return integral;
    152
129 153         case PHP_ROUND_HALF_EVEN:
130 154             if (fractional > 0.5) {
```

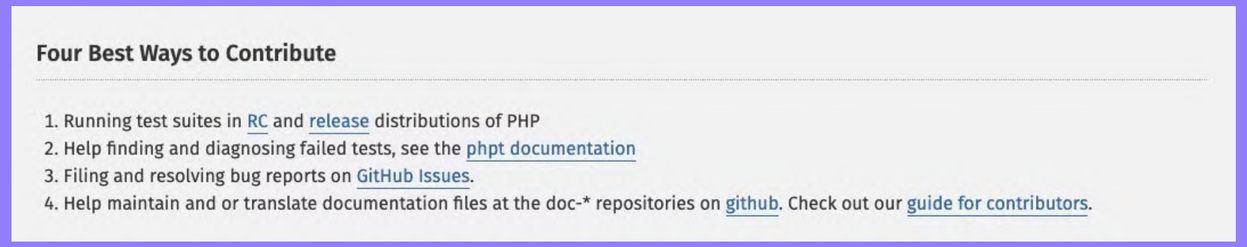
Try Pitch

Try Pitch

# Ways of contributing to PHP

Try Pitch

How to contribute to PHP language?



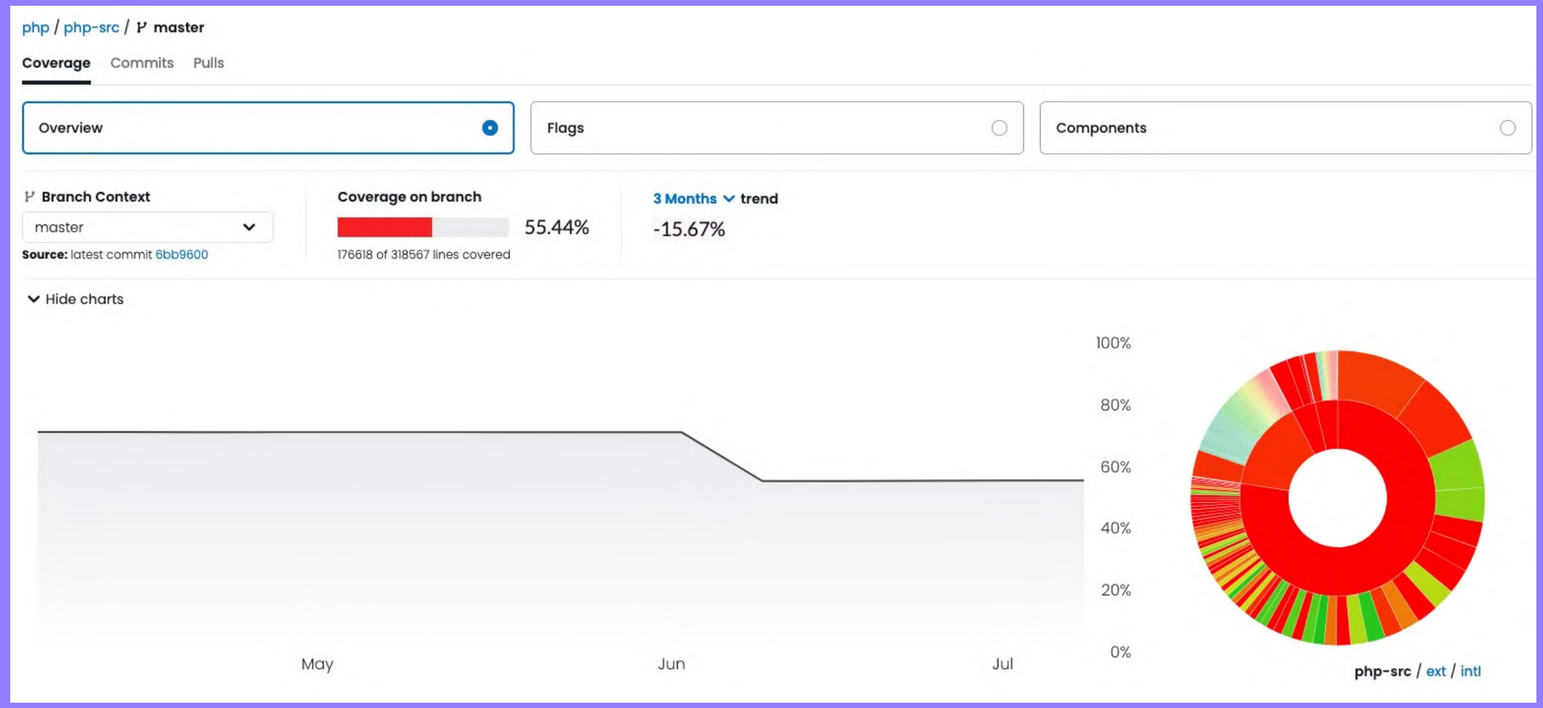
**Four Best Ways to Contribute**

1. Running test suites in RC and release distributions of PHP
2. Help finding and diagnosing failed tests, see the phpt documentation
3. Filing and resolving bug reports on GitHub Issues.
4. Help maintain and or translate documentation files at the doc-* repositories on github. Check out our guide for contributors.

https://www.php.net/get-involved

Try Pitch

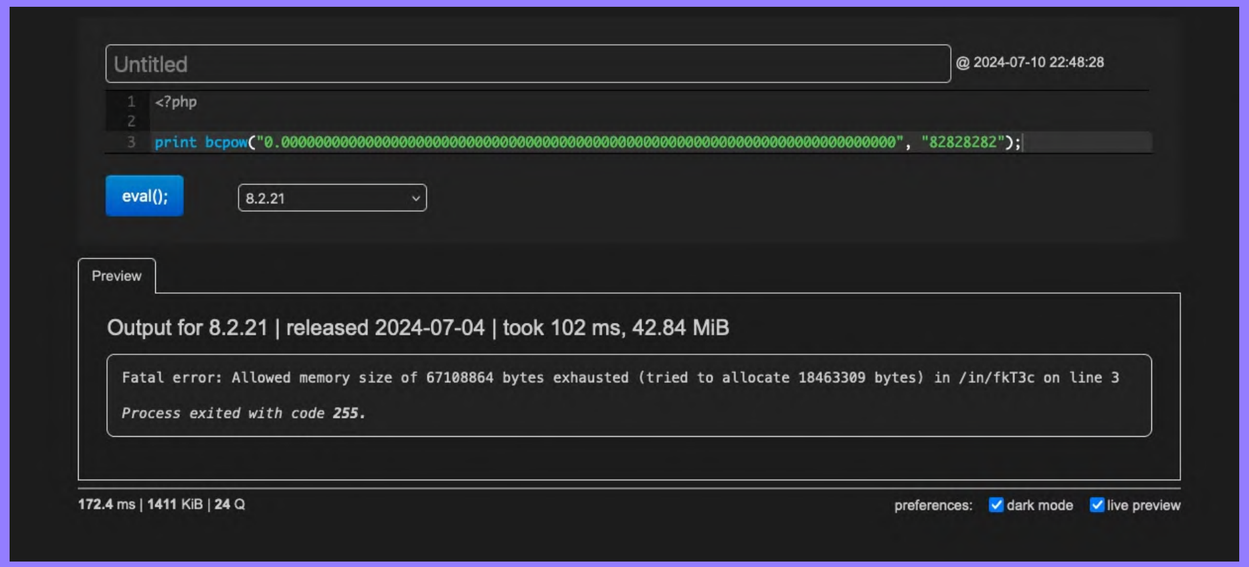
Improve code coverage



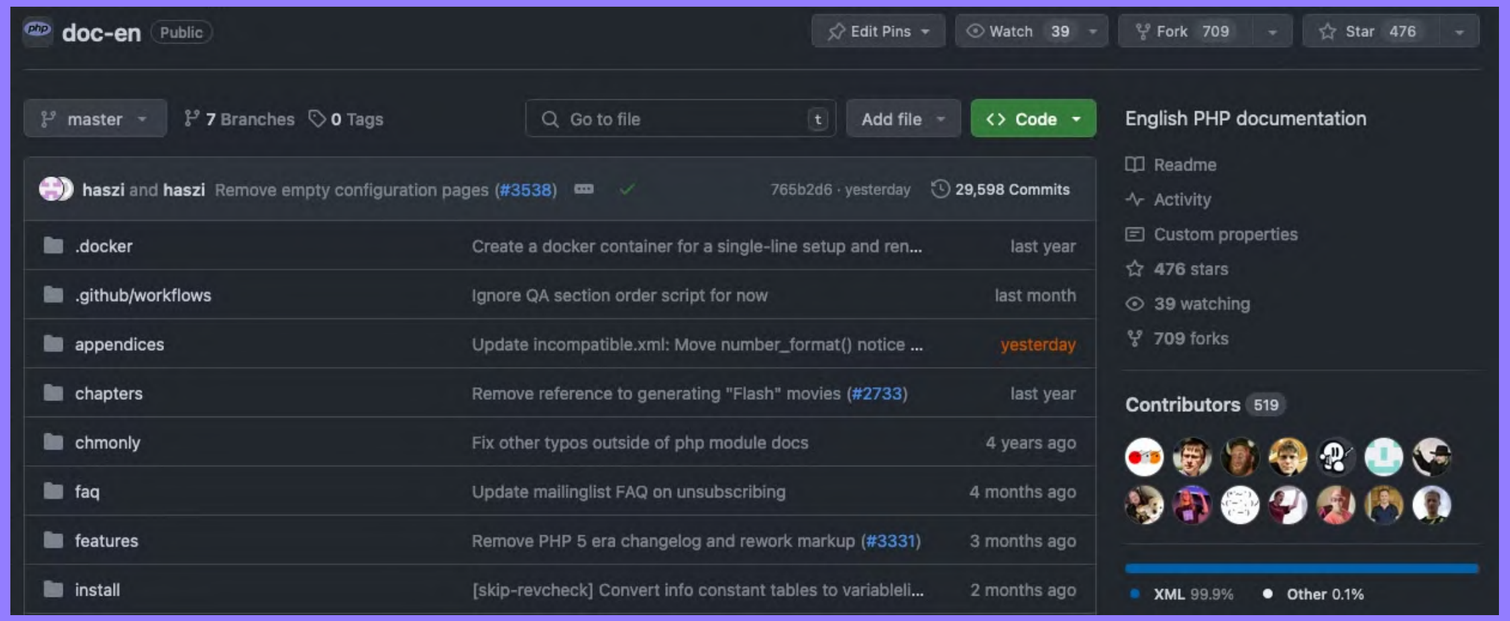

https://app.codecov.io/github/php/php-src/tree/master/ext

Try Pitch

Improve code coverage


Untitled
@ 2024-07-10 22:48:28
1 <?php
2
3 print bcpow("0.0000000000

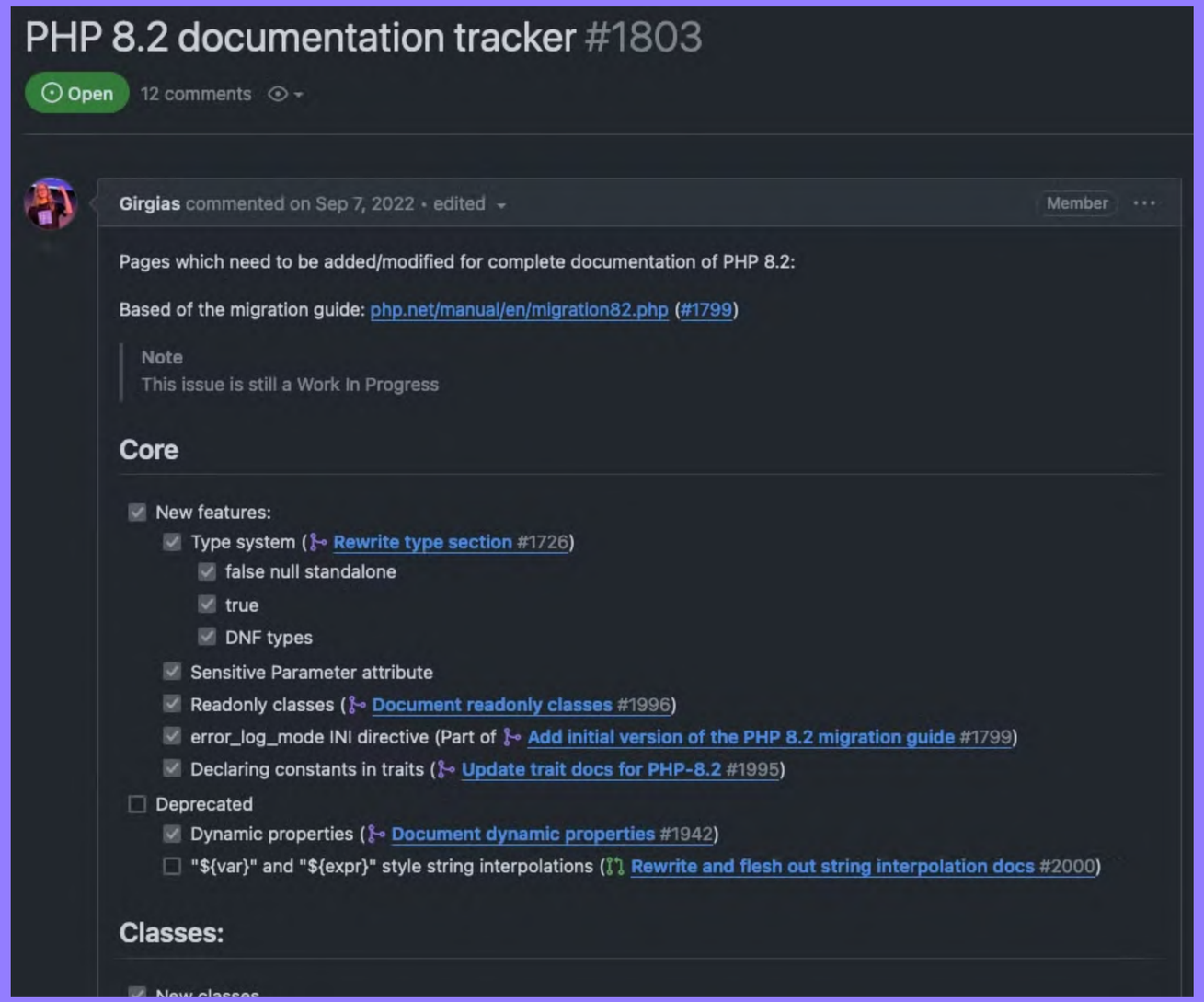
Improve PHP documentation




https://app.codecov.io/github/php/php-src/tree/master/ext

Try Pitch

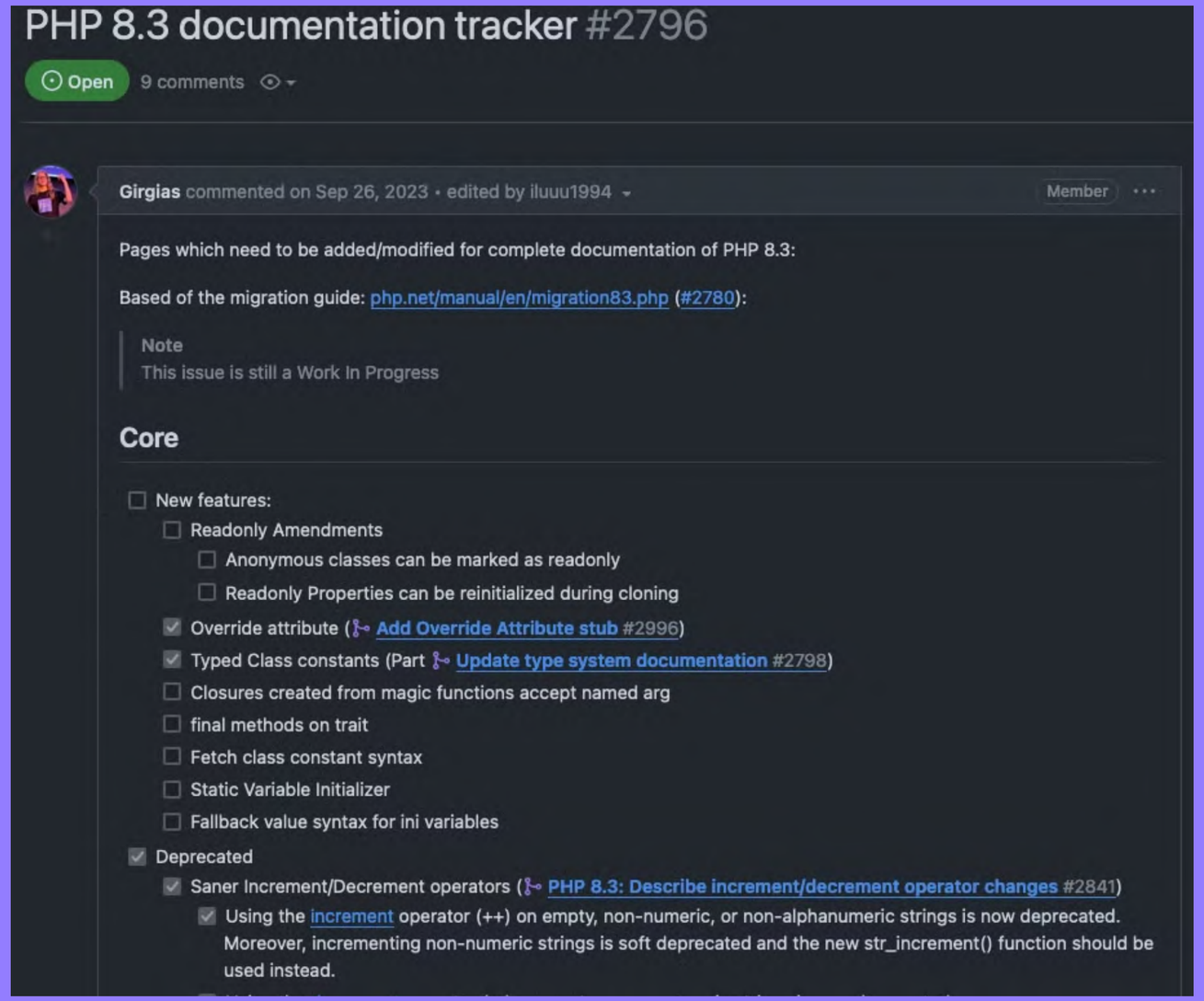
Improve PHP documentation




https://app.codecov.io/github/php/php-src/tree/master/ext

Try Pitch

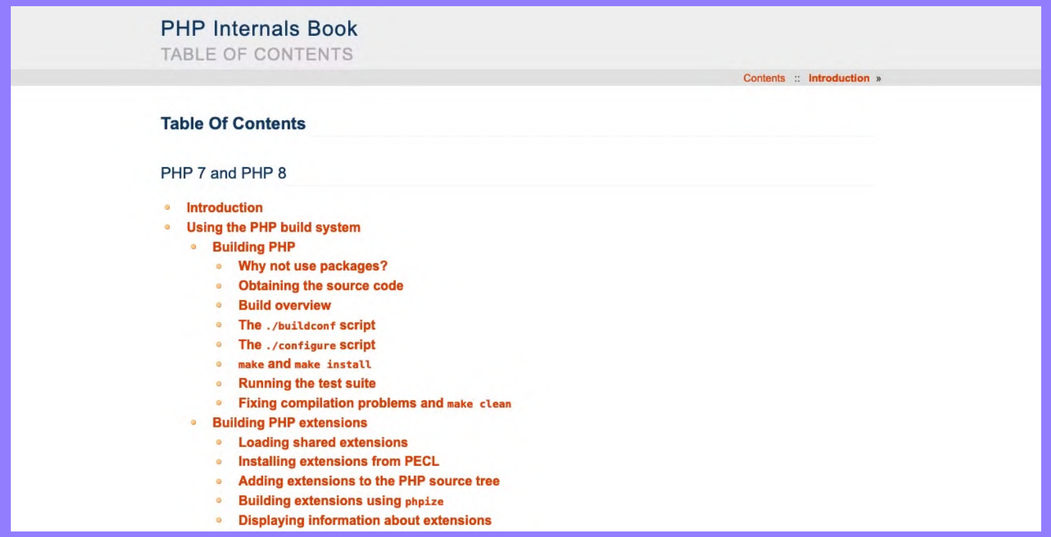
Improve PHP documentation




https://app.codecov.io/github/php/php-src/tree/master/ext

Try Pitch

# PHP Internals resources



PHP Internals Book

TABLE OF CONTENTS

Contents :: Introduction »

**Table Of Contents**

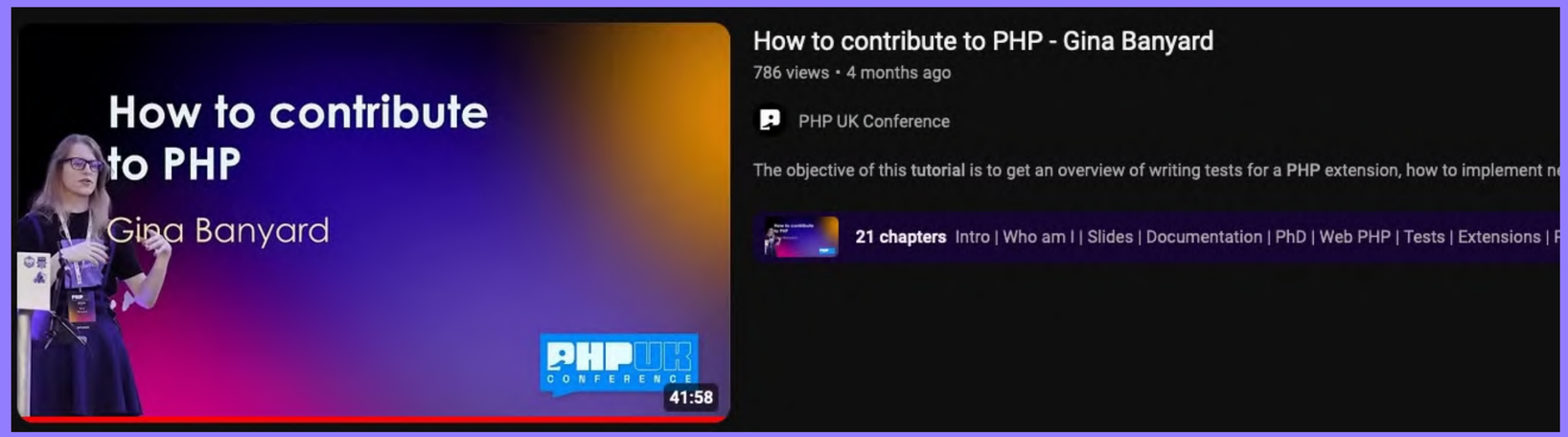
PHP 7 and PHP 8


- Introduction
- Using the PHP build system
  - Building PHP
    - Why not use packages?
    - Obtaining the source code
    - Build overview
    - The `./buildconf` script
    - The `./configure` script
    - `make` and `make install`
    - Running the test suite
    - Fixing compilation problems and `make clean`
  - Building PHP extensions
    - Loading shared extensions
    - Installing extensions from PECL
    - Adding extensions to the PHP source tree
    - Building extensions using `phpize`
    - Displaying information about extensions


https://www.phpinternalsbook.com/

Try Pitch

PHP Internals resources




https://www.youtube.com/watch?v=s-0pWt4gpFk

Try Pitch

# Thank you for listening

Adam Spychala
/jorgsowa

Try Pitch