

[doku.php](#)

```
<?php
2
3 /**
4  * Основной скрипт DokuWiki
5  *
6  * @license    GPL 2 (http://www.gnu.org/licenses/gpl.html)
7  * @author    Andreas Gohr <andi@splitbrain.org>
8  *
9  * @global Input $INPUT
10 */
11
12 используйте dokuwiki \ ChangeLog \ PageChangeLog ;
13 использовать dokuwiki \ Extension \ Event ;
14
15 // обновить версию сообщения — всегда используйте строку, чтобы избежать
    локализованных плавающих чисел!
16 $updateVersion = "56";
17
18 // xdebug_start_profiling();
19
20 if (!defined('DOKU_INC')) define('DOKU_INC', __DIR__ . '/');
21
22 // здесь определяются все глобальные переменные DokuWiki (необходимы в
    тестовых запросах, но также помогают отслеживать)
23 global $ACT, $INPUT, $QUERY, $ID, $REV, $DATE_AT, $IDX,
24         $DATE, $RANGE, $HIGH, $TEXT, $PRE, $SUF, $SUM, $INFO,
    $JSINFO;
25
26
27 if (isset($_SERVER['HTTP_X_DOKUWIKI_DO'])) {
28     $ACT = trim(strtolower($_SERVER['HTTP_X_DOKUWIKI_DO']));
29 } elseif (!empty($_REQUEST['idx'])) {
30     $ACT = 'index';
31 } elseif (isset($_REQUEST['do'])) {
32     $ACT = $_REQUEST['do'];
33 } else {
34     $ACT = 'show';
35 }
36
37 // загрузка и инициализация базовой системы
38 require_once(DOKU_INC . 'inc/init.php');
39
40 //импортировать переменные
41 $INPUT->set('id', str_replace("\xC2\xAD", '', $INPUT->str('id')));
    //мягкий перенос
42 $QUERY = trim($INPUT->str('q'));
43 $ID = getID();
44
45 $REV = $INPUT->int('rev');
```

```
46 $DATE_AT = $INPUT->str('at');
47 $IDX = $INPUT->str('idx');
48 $DATE = $INPUT->int('date');
49 $RANGE = $INPUT->str('range');
50 $HIGH = $INPUT->param('s');
51 if (empty($HIGH)) $HIGH = getGoogleQuery();
52
53 if ($INPUT->post->has('wikitext')) {
54     $TEXT = cleanText($INPUT->post->str('wikitext'));
55 }
56 $PRE = cleanText(substr($INPUT->post->str('prefix'), 0, -1));
57 $SUF = cleanText($INPUT->post->str('suffix'));
58 $SUM = $INPUT->post->str('summary');
59
60
61 //анализ DATE_AT
62 if ($DATE_AT) {
63     $date_parse = strtotime($DATE_AT);
64     if ($date_parse) {
65         $DATE_AT = $date_parse;
66     } else { // проверка метки времени UNIX
67         $date_parse = @date('Ymd', $DATE_AT);
68         if (!$date_parse || $date_parse === '19700101') {
69             msg(sprintf($lang['unable_to_parse_date'],
hsc($DATE_AT)));
70             $DATE_AT = null;
71         }
72     }
73 }
74
75 //проверка существующих $REV, относящихся к $DATE_AT
76 if ($DATE_AT) {
77     $pagelog = new PageChangeLog($ID);
78     $rev_t = $pagelog->getLastRevisionAt($DATE_AT);
79     if ($rev_t === '') {
80         //текущая редакция
81         $REV = null;
82         $DATE_AT = null;
83     } elseif ($rev_t === false) {
84         //страница не существует
85         $rev_n = $pagelog->getRelativeRevision($DATE_AT, +1);
86         msg(
87             sprintf(
88                 $lang['page_nonexist_rev'],
89                 dformat($DATE_AT),
90                 wl($ID, ['rev' => $rev_n]),
91                 dformat($rev_n)
92             )
93         );
94         $REV = $DATE_AT; // приведет к сообщению о том, что страница не
существует
```

```
95     } else {
96         $REV = $rev_t;
97     }
98 }
99
100 //сделать информацию о выбранной странице доступной
101 $INFO = pageinfo();
102
103 // обработка отладки
104 if ($conf['allowdebug'] && $ACT == 'debug') {
105     html_debug();
106     exit;
107 }
108
109 //отправлять 404 для отсутствующих страниц, если настроено или идентификатор
//имеет особое значение для ботов
110 if (
111     !$INFO['exists'] &&
112     ($conf['send404'] ||
preg_match('/^(robots\.txt|sitemap\.xml(\.gz)?|favicon\.ico|crossdomain
\.xml)$/ ', $ID)) &&
113     ($ACT == 'show' || (!is_array($ACT) && str_starts_with($ACT,
'export_'))))
114 ) {
115     header('HTTP/1.0 404 Not Found');
116 }
117
118 //подготовить хлебные крошки (инициализировать статическую переменную)
119 if ($conf['breadcrumbs']) breadcrumbs();
120
121 // проверка вверх по течению
122 checkUpdateMessages();
123
124 $tmp = []; // Нет данных о событии
125 Event::createAndTrigger('DOKUWIKI_STARTED', $tmp);
126
127 //закрыть сеанс
128 session_write_close();
129
130 //выполнить работу (выбирает, что делать, из глобальной среды)
131 act_dispatch();
132
133 $tmp = []; // Нет данных о событии
134 Event::createAndTrigger('DOKUWIKI_DONE', $tmp);
135
136 // xdebug_dump_function_profile(1);
137
```

«Подробности»

```
1<?php
2
3/**
4* Основной скрипт DokuWiki
5*
6* @license      GPL 2 ( http://www.gnu.org/licenses/gpl.html)
7* @автор       Андреас Гоп <andi@splitbrain.org>
8*
9* @global      Ввод $ ВВОД
10*/
11
12используйте   dokuwiki \ ChangeLog \ PageChangeLog ;
13использовать  dokuwiki \ Extension \ Event ;
14
15// обновить версию сообщения — всегда используйте строку, чтобы избежать
локализованных плавающих чисел!
16$ updateVersion = "56" ;
17
18// xdebug_start_profiling();
19
20если (! определено ( 'DOKU_INC' )) определить ( 'DOKU_INC' , __DIR__ . '/' );
21
22// здесь определяются все глобальные переменные DokuWiki (необходимы в тестовых
запросах, но также помогают отслеживать)
23глобальный $ ACT , $ INPUT , $ QUERY , $ ID , $ REV , $ DATE_AT , $ IDX ,
24    $ DATA , $ ДИАПАЗОН , $ ВЫСОКИЙ , $ ТЕКСТ , $ ПРЕД , $ СУФ , $ СУММА , $
ИНФОРМАЦИЯ , $ JSINFO ;
25
26
27если ( isset ( $ _SERVER [ 'HTTP_X_DOKUWIKI_DO' ] )) {
28    $ ACT = trim ( strtolower ( $ _SERVER [ 'HTTP_X_DOKUWIKI_DO' ] ));
29} elseif ( ! пусто ( $ _REQUEST [ 'idx' ] )) {
30    $ ACT = 'индекс';
31} elseif ( isset ( $ _REQUEST [ 'do' ] )) {
32    $ ACT = $ _REQUEST [ 'сделать' ];
33} еще {
34    $ ACT = 'показать';
35}
36
37// загрузка и инициализация базовой системы
38require_once ( DOKU_INC . ' inc / init.php ' );
39
40//импортировать переменные
41$ INPUT -> set ( 'id' , str_replace ( " \xC2 \xAD " , ' ' , $ INPUT -> str
( 'id' ) )); //мягкий перенос
42$ QUERY = trim ( $ INPUT -> str ( 'q' ) );
43$ ID = getID ();
44
45$ REV = $ INPUT -> int ( 'rev' );
46$ DATA_B = $ ВХОД -> стр ( 'в' );
47$ IDX = $ INPUT -> str ( 'idx' );
```

```
48$ ДАТА = $ ВВОД -> int ( 'дата' );
49$ ДИАПАЗОН = $ ВХОД -> стр ( 'диапазон' );
50$ HIGH = $ INPUT -> param ( 's' );
51если ( пусто ( $ HIGH ) ) $ HIGH = getGoogleQuery ();
52
53если ( $ INPUT -> post -> has ( 'wikitext' ) ) {
54    $ TEXT = cleanText ( $ INPUT -> post -> str ( 'wikitext' ) );
55}
56$ PRE = cleanText ( substr ( $ INPUT -> post -> str ( 'prefix' ), 0 , -1
));
57$ SUF = cleanText ( $ INPUT -> post -> str ( 'suffix' ) );
58$ SUM = $ INPUT -> post -> str ( 'summary' );
59
60
61//анализ DATE_AT
62если ( $ ДАТА_В ) {
63    $ date_parse = strtotime ( $ DATE_AT );
64    если ( $ date_parse ) {
65        $ ДАТА_В = $ дата_парс ;
66    } else { // проверка метки времени UNIX
67        $ date_parse = @date ( ' Ymd' , $ DATE_AT );
68        если ( !$ date_parse || $ date_parse === '19700101' ) {
69            msg ( sprintf ( $ lang [ 'unable_to_parse_date' ], hsc ( $
DATE_AT ) ) );
70            $ ДАТА_В = ноль ;
71        }
72    }
73}
74
75//проверка существующих $REV, относящихся к $DATE_AT
76если ( $ ДАТА_В ) {
77    $ pagelog = новый PageChangeLog ( $ ID );
78    $ rev_t = $ pagelog -> getLastRevisionAt ( $ DATE_AT );
79    если ( $ rev_t === '' ) {
80        //текущая редакция
81        $ REV = null ;
82        $ ДАТА_В = ноль ;
83    } elseif ( $ rev_t === false ) {
84        //страница не существует
85        $ rev_n = $ pagelog -> getRelativeRevision ( $ DATE_AT , +1 );
86        сообщение (
87            sprintf (
88                $ lang [ 'page_nonexist_rev' ],
89                dformat ( $ DATE_AT ),
90                wl ( $ ID , [ 'rev' => $ rev_n ] ),
91                dformat ( $ rev_n )
92            )
93        );
94        $ REV = $ DATE_AT ; // приведет к сообщению о том, что страница не
существует
95    } еще {
```

```
96      $ REV = $ rev_t ;
97  }
98}
99
100//сделать информацию о выбранной странице доступной
101$ ИНФОРМАЦИЯ = pageinfo ( );
102
103// обработка отладки
104если ( $ conf [ 'allowdebug' ] && $ ACT == 'debug' ) {
105    html_debug ( );
106    Выход ;
107}
108
109//отправлять 404 для отсутствующих страниц, если настроено или идентификатор имеет
особое значение для ботов
110если (
111    !$ ИНФОРМАЦИЯ [ 'существует' ] &&
112    ( $ conf [ 'send404' ] || preg_match (
    '/^(robots\.txt|sitemap\.xml(\.gz)?|favicon\.ico|crossdomain\.xml)$/ ' , $ ID
    ) ) &&
113    ( $ ACT == 'show' || (! is_array ( $ ACT ) && str_starts_with ( $ ACT ,
    'export_' )))
114) {
115    заголовок ( 'HTTP/1.0 404 Не найдено' );
116}
117
118//подготовить хлебные крошки (инициализировать статическую переменную)
119если ( $ conf [ 'хлебные крошки' ] ) хлебные крошки ( );
120
121// проверка вверх по течению
122checkUpdateMessages ( );
123
124$ tmp = []; // Нет данных о событии
125Событие :: createAndTrigger ( 'DOKUWIKI_STARTED' , $ tmp );
126
127//закрыть сеанс
128session_write_close ( );
129
130//выполнить работу (выбирает, что делать, из глобальной среды)
131act_dispatch ( );
132
133$ tmp = []; // Нет данных о событии
134Событие :: createAndTrigger ( 'DOKUWIKI_DONE' , $ tmp );
135
136// xdebug_dump_function_profile(1);
137
```

From:

<https://wwoss.ru/> - **worldwide open-source software**

Permanent link:

<https://wwoss.ru/doku.php?id=wiki:xref:dokuwiki:doku.php>

Last update: **2025/01/03 14:12**

