doku.php

```php
<?php

/**
 * Основной скрипт DokuWiki
 *
 * @license    GPL 2 (http://www.gnu.org/licenses/gpl.html)
 * @author     Andreas Gohr <andi@splitbrain.org>
 *
 * @global Input $INPUT
 */

используйте  dokuwiki \ ChangeLog \ PageChangeLog ;
использовать  dokuwiki \ Extension \ Event ;

// обновить версию сообщения — всегда используйте строку, чтобы избежать локализованных плавающих чисел!
$updateVersion = "56";

//  xdebug_start_profiling();

if (!defined('DOKU_INC')) define('DOKU_INC', __DIR__ . '/');

// здесь определяются все глобальные переменные DokuWiki (необходимы в тестовых запросах, но также помогают отслеживать)
global $ACT, $INPUT, $QUERY, $ID, $REV, $DATE_AT, $IDX,
       $DATE, $RANGE, $HIGH, $TEXT, $PRE, $SUF, $SUM, $INFO, $JSINFO;


if (isset($_SERVER['HTTP_X_DOKUWIKI_DO'])) {
    $ACT = trim(strtolower($_SERVER['HTTP_X_DOKUWIKI_DO']));
} elseif (!empty($_REQUEST['idx'])) {
    $ACT = 'index';
} elseif (isset($_REQUEST['do'])) {
    $ACT = $_REQUEST['do'];
} else {
    $ACT = 'show';
}

// загрузка и инициализация базовой системы
require_once(DOKU_INC . 'inc/init.php');

//импортировать переменные
$INPUT->set('id', str_replace("\xC2\xAD", '', $INPUT->str('id')));
//мягкий перенос
$QUERY = trim($INPUT->str('q'));
$ID = getID();

$REV = $INPUT->int('rev');
```

```
$DATE_AT = $INPUT->str('at');
$IDX = $INPUT->str('idx');
$DATE = $INPUT->int('date');
$RANGE = $INPUT->str('range');
$HIGH = $INPUT->param('s');
if (empty($HIGH)) $HIGH = getGoogleQuery();

if ($INPUT->post->has('wikitext')) {
    $TEXT = cleanText($INPUT->post->str('wikitext'));
}
$PRE = cleanText(substr($INPUT->post->str('prefix'), 0, -1));
$SUF = cleanText($INPUT->post->str('suffix'));
$SUM = $INPUT->post->str('summary');


//анализ DATE_AT
if ($DATE_AT) {
    $date_parse = strtotime($DATE_AT);
    if ($date_parse) {
        $DATE_AT = $date_parse;
    } else { // проверка метки времени UNIX
        $date_parse = @date('Ymd', $DATE_AT);
        if (!$date_parse || $date_parse === '19700101') {
            msg(sprintf($lang['unable_to_parse_date'],
hsc($DATE_AT)));
            $DATE_AT = null;
        }
    }
}

//проверка существующих $REV, относящихся к $DATE_AT
if ($DATE_AT) {
    $pagelog = new PageChangeLog($ID);
    $rev_t = $pagelog->getLastRevisionAt($DATE_AT);
    if ($rev_t === '') {
        //текущая редакция
        $REV = null;
        $DATE_AT = null;
    } elseif ($rev_t === false) {
        //страница не существует
        $rev_n = $pagelog->getRelativeRevision($DATE_AT, +1);
        msg(
            sprintf(
                $lang['page_nonexist_rev'],
                dformat($DATE_AT),
                wl($ID, ['rev' => $rev_n]),
                dformat($rev_n)
            )
        );
        $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 доступнойе
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»

```
<?php

/**
* Основной скрипт DokuWiki
*
* @license     GPL 2 ( http://www.gnu.org/licenses/gpl.html)
* @автор    Андреас Гор <andi@splitbrain.org>
*
* @global  Ввод $ ВВОД
*/

используйте  dokuwiki \ ChangeLog \ PageChangeLog ;
использовать  dokuwiki \ Extension \ Event ;

// обновить версию сообщения — всегда используйте строку, чтобы избежать локализованных плавающих чисел!
$ updateVersion = "56" ;

// xdebug_start_profiling();

если (! определено ( 'DOKU_INC' )) определить ( 'DOKU_INC' , __DIR__ . '/' );

// здесь определяются все глобальные переменные DokuWiki (необходимы в тестовых запросах, но также помогают отслеживать)
глобальный $ ACT , $ INPUT , $ QUERY , $ ID , $ REV , $ DATE_AT , $ IDX ,
    $ ДАТА , $ ДИАПАЗОН , $ ВЫСОКИЙ , $ ТЕКСТ , $ ПРЕД , $ СУФ , $ СУММА , $ ИНФОРМАЦИЯ , $ JSINFO ;


если ( isset ($ _SERVER [ 'HTTP_X_DOKUWIKI_DO' ])) {
    $ ACT = trim ( strtolower ($ _SERVER [ 'HTTP_X_DOKUWIKI_DO' ]));
} elseif (! пусто ($ _REQUEST [ 'idx' ])) {
    $ ACT = 'индекс';
} elseif ( isset ($ _REQUEST [ 'do' ])) {
    $ ACT = $ _REQUEST [ 'сделать'];
} еще {
    $ ACT = 'показать';
}

// загрузка и инициализация базовой системы
require_once ( DOKU_INC . ' inc / init.php ' );

//импортировать переменные
$ INPUT -> set ( 'id' , str_replace ( " \xC2 \xAD " , '' , $ INPUT -> str ( 'id' ))); //мягкий перенос
$ QUERY = trim ($ INPUT -> str ( 'q' ));
$ ID = getID ();

$ REV = $ INPUT -> int ( 'rev' );
$ ДАТА_В = $ ВХОД -> стр ( 'в' );
$ 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      спринтф (
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
```