

```
<?php

/*! \mainpage Artikkelihallintaohjelma
 *
 * \section intro_sec Introduction
 *
 * Tämän on artikkelienhallintaohjelmiston lahdekoodin dokumentoitu.
 *
 * \section outro_sec Author
 * Daniel Tisza, 2011
 * Jussi Isosävi, 2012
 */

/**
 * Copyright: @author Daniel Tisza, 2011
 */

session_start();
srand( time() ); // Seed random generator
// Force HTTPS, not implemented

/*!
 * \class Root
 */
class Root {
}

/*!
 * \class Input
 * \brief syöttökenttä
 *
 * \details lomakkeella yksi syötekenttä
 */

class Input extends Root
{
    var $name;
    var $type;
    var $value;
    var $size;
    var $errmsg;

    /*!
     * \brief luomisifunktio
     * \details luodaan uusi syötekenttä
     *
     * \param name käyttäjälle näytettävä nimi
     * \param type käyttäjälle näytettävä tyyppi
     * (numero/teksti)
     * \param value käyttäjälle näytettävä oletusarvo
     * \param size kentän koko
     * \param errmsg käyttäjälle näytettävä virheteksti,
     * jos arvo ei kelpaa
     */
    function Input( $name, $type, $value, $size, $errmsg )
    {
        $this->name = $name;
        $this->type = $type;
        $this->value = $value;
        $this->size = $size;
        $this->errmsg = $errmsg;
    }
}
```

```

90
91
92
93      /*!
94      * \brief      Luo HTML Input tägin.
95      * \details    Luo muuttujien arvoilla html-tägin.
96      *
97      * \return     palauttaa html-tägin tekstinä.
98      */
99      function html( )
100     {
101         $html = '<input name="'.md5( $this->name ).'".
102                ' type="'. $this->type. '".
103                ' size="'. $this->size. '";
104
105         if( strcmp( $this->type, 'file' ) == 0 )
106         {
107             $html .= '>'. $this->errmsg;
108         }
109         else
110         {
111             $html .= ' value="'. $this->value. '>';
112
113             if ( $this->value == '' )
114             {
115                 // ilmoitetaan virheviesti jos kenttä on tyhjä
116                 $html .= $this->errmsg;
117             }
118         }
119
120         return $html;
121     }
122
123     /*!
124     * \brief      muotoilee syötekentän tekstiksi.
125     * \brief      Tekstissä on ensin nimi ja arvo
126     *              kaksoispisteellä erotettuna.
127     *
128     * \return     palauttaa syötekentän tekstinä
129     */
130     function txt( )
131     {
132         if( strcmp( $this->type, 'submit' ) == 0 ||
133            strcmp( $this->type, 'file' ) == 0 )
134         {
135             return;
136         }
137
138         $txt = $this->name.' : '. $this->value;
139
140         return $txt;
141     }
142
143
144     /*!
145     * \brief      Tiedosto-syötekentän tiedoston nimi.
146     * \details    Tiedosto-tyyppisessä syötekentässä ladatun tiedoston nimi.
147     *
148     * \return     palauttaa väliaikaisen tiedostonimen.
149     */
150     function filename( )
151     {
152         $filename = '';
153
154         if( strcmp( $this->type, 'file' ) == 0 )
155         {
156             $filename = $_FILES[ md5( $this->name ) ][ 'tmp_name' ];
157         }
158
159         return $filename;
160     }
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176

```

```
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264

/#!/
* \brief      Kerää tiedot lomakkeen syötekentistä
* \details    Kerää tiedot lomakkeen syötekentistä
*
* \return     Palautetaan 1, jos kentän arvo kelpaa.
*             muussa tapauksessa 0.
*/

function verified_fill( )
{
    if( strcmp( $this->type, 'submit' ) == 0 )
        return;

    if( strcmp( $this->type, 'file' ) == 0 )
    {
        $this->value = $_FILES[ md5( $this->name ) ]['name'];
    }
    else
    {
        $post = $_POST[ md5( $this->name ) ];
        $post = trim( $post );
        $post = substr( $post, 0, $this->size );
        $post = escapeshellcmd( $post );

        $this->value = $post;
    }

    if ( $this->value == '' ) /** jos arvo on tyhjä */
    {
        return 0; /** huono arvo */
    }

    return 1; /** arvo kelpaa */
}

/#!/
* \class Form
* \brief      Tietojensyöttölomake
* \details    kerää tiedot lomakkeen syötekentistä
*
* \details    lomake sisältää syöttökenttiä ja tunnistekentän
*
*/
class Form extends Root
{
    var $action;
    var $method;
    var $enctype;
    var $inputs;
    var $captcha_input;
    var $submit_input;
    var $captcha_img_url;
```

```

266
267
268
269
270
271  /*!
272  * \brief      luomisifunktio
273  * \details    luodaan uusi lomake
274  *
275  * \param      action          Osoite, johon lomake lähetetään
276  * \param      method          Tapa, jolla lomake lähetetään (GET/POST)
277  * \param      enctype         muoto, jossa lomake lähetetään palvelimelle
278  *                                     (multipart/form-data)
279  * \param      captcha_img_url Osoite, mistä tunnistekuva haetaan
280  */
281  function Form( $action, $method, $enctype, $captcha_img_url )
282  {
283      $this->action = $action;
284      $this->method = $method;
285      $this->enctype = $enctype;
286      $this->inputs = array( );
287      $this->captcha_input = new Input( 'Verification Code in Image', '', '', 5 );
288      $this->submit_input = new Input( '', 'submit', 'Submit', 10 );
289      $this->captcha_img_url = $captcha_img_url;
290  }
291
292  /*!
293  * \brief      Syötekentän lisäys-funktio
294  * \details    Lisätään syötekenttä lomakkeelle
295  *             edellisten perään taulukkoon
296  *
297  * \param      input          Lisättävä syötekenttä
298  */
299  function add_input( $input )
300  {
301      $this->inputs[] = $input;
302  }
303
304
305  /*!
306  * \brief      Lomakkeen täytön tarkistaminen.
307  * \details    Käydään lläpi kaikki lomakkeen tiedot.
308  *
309  * \return     Hyväksytyt täytön tunnisteissa
310  */
311  function verified_fill( )
312  {
313      $valid_fill = 1;
314
315      foreach( $this->inputs as $key => $val )
316      {
317          if ( $val->verified_fill( ) != 1 )
318          {
319              // Joku kenttä on väärin -> lomake väärin
320              $valid_fill = 0;
321          }
322      }
323
324      return $valid_fill;
325  }
326
327  /*!
328  * \brief      Muodosta lomakkeen aloitustägi.
329  * \details    Muodosta lomakkeen aloitustägi.
330  *
331  * \return     Palauttaa html-tägin tekstinä.
332  */
333  function begin_html( )
334  {
335      $html = '<form action="'. $this->action. '"'.
336              ' method="'. $this->method. '"'.
337              ' enctype="'. $this->enctype. '">';
338
339      return $html;
340  }
341
342
343
344
345
346
347
348
349
350
351
352

```

```

354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440

```

```

/*!
 * \brief      Muodosta lomakkeen lopetustägi.
 * \details    Lisää tunnistekuvan ja lopetustägin.
 *
 * \return     Palauttaa html-tägin tekstinä.
 */
function end_html( )
{
    $html = '<br>'.
            $this->captcha_input->name.' ' . $this->captcha_input->html( ).'<br>'.
            $this->submit_input->html( ).
            '</form>';

    return $html;
}

/*!
 * \brief      Muodota syötekenttien tägit.
 * \details    Syötekentät laitetaan taulukkoon.
 *
 * \return     syötekenttien tägit tekstinä.
 */
function inputs_html( )
{
    $html = '<table>';

    foreach ( $this->inputs as $key => $val )
    {
        $html .= '<tr><td>'. $val->name.'</td><td>'. $val->html( ).'</td></tr>';
    }

    $html .= '</table>';

    return $html;
}

/*!
 * \brief      Kerää lomakkeen tiedot tesktinä.
 * \details    Kerää lomakkeen tiedot tesktinä ja
 *             jokainen syötekenttä omalla rivillään.
 *
 * \return     Palauttaa lomakkeen tiedot tekstinä.
 */
function inputs_txt( )
{
    $txt = '';

    foreach ( $this->inputs as $key => $val )
    {
        $txt .= $val->txt( )."\\r\\n";
    }

    return $txt;
}

/*!
 * \brief      Kerää lomakkeen Tiedosto-syötekenttien tiedostonimet.
 * \details    Tiedosto-tyyppisessä syötekentässä ladatun tiedoston nimi.
 *
 * \return     palauttaa taulukon väliaikeisista tiedostonimistä.
 */
function inputs_filenames( )
{
    $filenames = array( );

    foreach ( $this->inputs as $key => $val )
    {
        $filename = $val->filename( );

        if ( $filename != '' )
        {
            $filenames[] = array('tmp_name' => $filename, 'name' => $val->value );
        }
    }

    return $filenames;
}

```

```

442
443
444
445 /*!
446 * \brief      Tunnisteen tarkistus-funktio
447 * \details    Tarkistaa tunnisteen oikeellisuuden.
448 *
449 * \return     Palauttaa 1, jos tunniste on oikein
450 *             ja muissa tapauksissa 0.
451 */
452
453 function verify_captcha( )
454 {
455     $this->captcha_input->verified_fill( );
456     $entered_captcha = $this->captcha_input->value;
457     $this->captcha_input->value = '';
458
459     if ( $entered_captcha != '' )
460     {
461         $captcha_hash = md5( $entered_captcha.$_SESSION['captchasalt'] );
462
463         if ( strcmp( $_SESSION['captcha_hash'], $captcha_hash ) == 0 )
464         {
465             unset( $_SESSION['captcha_hash'] );
466             unset( $_SESSION['captchasalt'] );
467             return 1;
468         }
469     }
470
471     unset( $_SESSION['captcha_hash'] );
472     unset( $_SESSION['captchasalt'] );
473     return 0;
474 }
475 }
476
477 /**
478 * @brief      Tunnistekuvassa olevan tekstin generointi
479 * @return     Arvottu tunnisteteksti
480 */
481
482 function captcha_text( )
483 {
484     /* määritellään tunnisteen arvot */
485     $chars = array( 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'm',
486                   'n', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'y', 'z',
487                   '2', '3', '4', '5', '6', '8', '9', '@', '#', '%', '?',
488                   'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'J', 'K', 'L', 'M',
489                   'N', 'P', 'R', 'S', 'T', 'U', 'V', 'W', 'Y', 'Z' );
490
491     $text = '';
492
493     /* arvojen arpominen */
494     for ( $i = 0; $i < 5; $i++ )
495     {
496         $text .= $chars[ rand( 0, count( $chars ) - 1 ) ];
497     }
498
499     return $text; /* palautetaan saatu arvo */
500 }
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528

```

```

530
531
532
533 /**
534 * @brief Tulostaa sivulle Tunnistekuvan.
535 */
536 function captcha_image( )
537 {
538     header( 'Content-type: image/jpeg' );
539
540     $text = captcha_text( );
541     $salt = rand();
542     $_SESSION['captchasalt'] = $salt;
543     $_SESSION['captchahash'] = md5( $text.$salt ); // Salted hash of solution
544
545     /* määritetään tunnisteiden leveys, korkeus, */
546     $w = 30;
547     $h = 30;
548     $charw = 2.3 * $w;
549     $charh = 2.3 * $h;
550     $zw = 400;
551     $zh = 150;
552
553     $imgzoom = imagecreatetruecolor( $zw, $zh );
554     $zoomwhite = imagecolorallocate( $imgzoom, 255, 255, 255 );
555
556     for ( $i = 0; $i < 5; $i++ ) // Taustan värit, kirjaimien väri
557     {
558         $img = imagecreatetruecolor( $w, $h );
559         $color = imagecolorallocate( $img, 255, 255, 255 );
560
561         imagechar( $img, 5, 0, 0, $text[ $i ], $color );
562         $imgrot = imagerotate( $img, rand( -30, 30 ), 0 );
563         imagedestroy( $img );
564
565         $x = 8 + rand( 60, 65 ) * $i;
566         $y = 5 + rand( 0, 70 );
567
568         imagecopyresampled( $imgzoom, $imgrot, $x, $y, 0, 0, $charw, $charh, $w, $h );
569         imagedestroy( $imgrot );
570     }
571
572     for ( $i = 0; $i < 5; $i++ ) // Taustan viivat
573     {
574         imageline( $imgzoom, rand( 0, $zw ), rand( 0, $zh ),
575                 rand( 0, $zw ), rand( 0, $zh ), $zoomwhite );
576     }
577
578     for ( $i = 0; $i < 3; $i++ ) // Taustan kaaret
579     {
580         imagearc( $imgzoom, rand( 0, $zw ), rand( 0, $zh ),
581                rand( 0, $zw ), rand( 0, $zh ),
582                rand( 0, 360 ), rand( 0, 360 ), $zoomwhite );
583     }
584
585     imagejpeg( $imgzoom, NULL, 15 );
586     imagedestroy( $imgzoom );
587 }
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616

```

```

618
619
620
621 /**
622  * @brief      Muotoilee ja lähettää sähköpostin.
623  *
624  * @param      from      lähettäjän osoite
625  * @param      to        vastaanottajan osoite
626  * @param      subject   aihe
627  * @param      msg       viesti
628  *
629  * \return Palauttaa lähetys-funktion paluuarvon.
630  */
631
632 function format_send_email( $from, $to, $subject, $msg )
633 {
634     $headers = "From: $from\n".
635               "Reply-To: $from\n".
636               "Return-Path: $from\n".
637               "MIME-Version: 1.0\n".
638               "Content-type: text/plain; charset=iso-8859-1\n";
639
640     $msg = wordwrap( $msg, 69, "\r\n" );
641
642     $res = mail( $to, $subject, $msg, $headers );
643
644     return $res;
645 }
646
647 /**
648  * @brief      Muotoilee ja lähettää sähköpostin liitetiedoston kanssa.
649  *
650  * @param      from      lähettäjän osoite
651  * @param      to        vastaanottajan osoite
652  * @param      subject   aihe
653  * @param      filename  tiedostonimi
654  * @param      attachedfilename  liitetiedoston nimi
655  *
656  * \return Palauttaa lähetys-funktion paluuarvon.
657  */
658 function send_email_file( $from, $to, $subject, $filename, $attachedfilename )
659 {
660     $headers = "From: $from\n".
661               "Reply-To: $from\n".
662               "Return-Path: $from\n".
663               "MIME-Version: 1.0\n".
664               "Content-type: application/zip; name="'. $attachedfilename.''. "\n".
665               "Content-Transfer-Encoding: base64\n".
666               "Content-Disposition: attachment\n";
667
668     $msg = chunk_split(base64_encode( file_get_contents( $filename ) ) );
669
670     $res = mail( $to, $subject, $msg, $headers );
671
672     return $res;
673 }
674
675 /**
676  * @brief      Turha funktio?
677  *
678  * @param      originalname  ?
679  * @param      tmpname       ?
680  */
681 function add_extension( $originalname, $tmpname )
682 {
683 }
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704

```



```

706
707
708
709 /**
710  * @brief      pakkaa annetun tiedoston.
711  *
712  * @param      filename      pakattavan tiedoston nimi.
713  *
714  * \return     paluttaa pakatun tiedoston nimen.
715  */
716 function temporary_zip_file( $filename )
717 {
718     $tmpdir = '/tmp/';
719     $random = md5( date( 'r', time() ) );
720     $tmpzip = $tmpdir.$random.'.zip';
721
722     exec( 'zip '.$tmpzip.' '.$filename ); //luo .zip-tiedoston
723
724     $f = fopen( $tmpzip, 'r' ); //todentaa .zip-tiedoston
725
726     if ( $f )
727     {
728         fclose( $f );
729     }
730     else
731     {
732         return 0;
733     }
734
735     return $tmpzip;
736 }
737
738 if ( $_GET['captcha'] == '1' )
739 {
740     captcha_image( );
741     exit;
742 }
743
744 $captcha_img_url = $_SERVER['SCRIPT_NAME'].'?captcha=1&nocache='.md5( rand() );
745
746 $form = new Form( $_SERVER['SCRIPT_NAME'], 'post',
747                 'multipart/form-data', $captcha_img_url );
748
749 $form->add_input( new Input( 'First Name', '', '', 20, 'Must be non-empty' ) );
750 $form->add_input( new Input( 'Last Name', '', '', 20, 'Must be non-empty' ) );
751 $form->add_input( new Input( 'Email Address', '', '', 20, 'Must be non-empty' ) );
752 $form->add_input( new Input( 'Institution', '', '', 40, 'Must be non-empty' ) );
753 $form->add_input( new Input( 'Publication Title', '', '', 40, 'Must be non-empty' ) );
754 $form->add_input( new Input( 'Publication File', 'file', '', 30, 'Must be non-empty' ) );
755
756 echo '<html>';
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792

```

794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858

```
if ( $form->verified_fill( ) && $form->verify_captcha( ) )
{
    $from = 'k83660@student.uwasa.fi'; /* mistä lähetään*/
    $to = 'jussiisosavi@gmail.com'; /* minne lähetetään*/

    $res = format_send_email( $from, $to, 'Information', $form->inputs_txt( ) );

    if ( $res )
    {
        echo 'Information Submitted.<br>';
    }
    else
    {
        echo 'Information Submission Failed.<br>';
    }

    foreach ( $form->inputs_filenames( ) as $key => $val )
    {
        $file_extension = pathinfo( $val['name'], PATHINFO_EXTENSION );

        $filename = $val['tmp_name'].'.'.$file_extension;

        if ( rename( $val['tmp_name'], $filename ) != true )
        {
            echo 'Unable to use file extension. File not submitted.<br>';
            continue;
        }

        $tmpzip = temporary_zip_file( $filename );

        if ( $tmpzip === 0 )
        {
            echo 'Unable to compress file.';
            continue;
        }

        $res = send_email_file( $from, $to, 'Publication File',
                               $tmpzip, 'Publication.zip' );

        if ( $res )
        {
            echo 'File Submitted.<br>';
        }
        else
        {
            echo 'File Not Submitted.<br>';
        }
    }
}
else
{
    echo $form->begin_html( ),
        $form->inputs_html( ),
        $form->end_html( );
}

echo '</html>';
?>
```