

Kartierungshilfsmittel

php Script Funktionen zur Koordinatentransformation

Für die hier gezeigten 8 Funktionen zur Koordinatentransformation wird die Existenz der folgenden Variablen vorausgesetzt

```
global $rw, $hw, $lp, $bp, $lw, $bw, $zone, $ew, $nw, $raster, $ew2, $nw2;
```

Diese haben folgende Bedeutung

\$rw, \$hw	Rechts-, Hochwert im deutschen Gauss-Krüger-System
\$lp, \$bp	Geographische Länge und Breite auf dem Bessel-Ellipsoid (Potsdam Datum) in Grad mit Dezimalpunkt (nicht Komma)
\$lw, \$bw	Geographische Länge und Breite im auf dem Internationalen Ellipsoid (WGS84 Datum), westliche Länge und südliche Breite negativ
\$zone, \$ew, \$nw	UTM Gitterzone (2 Ziffern für die Längenzone und ein Buchstabe für das Breitenband, Beispiel: 32U), 7 stelliger Ost- und Nordwert im UTM System
\$raster, \$ew2, \$nw2	UTMREF Raster (UTM Gitterzone und Kennung aus 2 Buchstaben für das 100 km × 100 km Feld, Beispiel: 32UMT), 5 stelliger Ost- und Nordwert

Umrechnungen im WGS84 System

Die beiden folgenden Funktionen rechnen im WGS84 Kartenbezugssystem (world geodetic system 84) geographische Koordinaten in UTM Koordinaten um und umgekehrt.

Mit der Funktion geo2utm.php lassen sich geographische Längen und Breiten in UTM Zone, Ost- und Nordwert umrechnen:

```
function geo2utm()
{
/* Copyright (c) 2006, HELMUT H. HEIMEIER
Permission is hereby granted, free of charge, to any person obtaining a
copy of this software and associated documentation files (the "Software"),
to deal in the Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense,
and/or sell copies of the Software, and to permit persons to whom the
Software is furnished to do so, subject to the following conditions:
The above copyright notice and this permission notice shall be included
in all copies or substantial portions of the Software.*/

/* Die Funktion wandelt geographische Koordinaten in UTM Koordinaten
um. Geographische Länge lw und Breite bw müssen im WGS84 Datum
gegeben sein. Berechnet werden UTM Zone, Ostwert ew und Nordwert nw.

global $rw, $hw, $lp, $bp, $lw, $bw, $zone, $ew, $nw, $raster, $ew2, $nw2;

//Geographische Länge lw und Breite bw im WGS84 Datum
if ($lw == "" || $bw == "")
{
$zone = "";
$ew = "";
$nw = "";
return;
}
if ($lw <= -180 || $lw > 180 || $bw <= -80 || $bw >= 84)
{
echo("Werte nicht im Bereich des UTM Systems\n"+
```

```

"-180° <= LW < +180°, -80° < BW < 84° N");
$zone = "";
$ew = "";
$nw = "";
return;
}
$lw = doubleval($lw);
$bw = doubleval($bw);

//WGS84 Datum
//Große Halbachse a und Abplattung f
$a = 6378137.000;
$f = 3.35281068e-3;
$b_sel = 'CDEFGHJKLMNPQRSTUVWXYZ';

//Polkrümmungshalbmesser c
$c = $a/(1-$f);

//Quadrat der zweiten numerischen Exzentrizität
$ex2 = (2*$f-$f*$f)/((1-$f)*(1-$f));
$ex4 = $ex2*$ex2;
$ex6 = $ex4*$ex2;
$ex8 = $ex4*$ex4;

//Koeffizienten zur Berechnung der Meridianbogenlänge
$e0 = $c*(pi()/180)*(1 - 3*$ex2/4 + 45*$ex4/64 - 175*$ex6/256 + 11025*$ex8/16384);
$e2 = $c*(- 3*$ex2/8 + 15*$ex4/32 - 525*$ex6/1024 + 2205*$ex8/4096);
$e4 = $c*(15*$ex4/256 - 105*$ex6/1024 + 2205*$ex8/16384);
$e6 = $c*(- 35*$ex6/3072 + 315*$ex8/12288);

//Längenzone lz und Breitenzone (Band) bz
$lzn = intval(($lw+180)/6) + 1;
$lz = "$lzn";
if ($lzn < 10) $lz = "0" + $lzn;
$bd = intval(1 + ($bw + 80)/8);
$bz = substr($b_sel,$bd-1,1);

//Geographische Breite in Radianen br
$br = $bw * pi()/180;

$tan1 = tan($br);
$tan2 = $tan1*$tan1;
$tan4 = $tan2*$tan2;

$cos1 = cos($br);
$cos2 = $cos1*$cos1;
$cos4 = $cos2*$cos2;
$cos3 = $cos2*$cos1;
$cos5 = $cos4*$cos1;

$etasq = $ex2*$cos2;

//Querkrümmungshalbmesser nd
$nd = $c/sqrt(1 + $etasq);

//Meridianbogenlänge g aus gegebener geographischer Breite bw
$g = ($e0*$bw) + ($e2*sin(2*$br)) + ($e4*sin(4*$br)) + ($e6*sin(6*$br));

//Längendifferenz dl zum Bezugsmeridian lh
$lh = ($lzn - 30)*6 - 3;
$d1 = ($lw - $lh)*pi()/180;
$d12 = $d1*$d1;
$d14 = $d12*$d12;
$d13 = $d12*$d1;
$d15 = $d14*$d1;

//Maßstabsfaktor auf dem Bezugsmeridian bei UTM Koordinaten m = 0.9996
//Nordwert nw und Ostwert ew als Funktion von geographischer Breite und Länge

```

```

if ( $bw < 0 ) {
    $nw = 10e6 + 0.9996*($g + $nd*$cos2*$tan1*$dl2/2 +
        $nd*$cos4*$tan1*(5-$tan2+9*$etasq)*$dl4/24);
}
else {
    $nw = 0.9996*($g + $nd*$cos2*$tan1*$dl2/2 +
        $nd*$cos4*$tan1*(5-$tan2+9*$etasq)*$dl4/24);
}
$ew = 0.9996*( $nd*$cos1*$dl + $nd*$cos3*(1-$tan2+$etasq)*$dl3/6 +
    $nd*$cos5*(5-18*$tan2+$tan4)*$dl5/120) + 500000;

$zone = $lz . $bz;

$nk = $nw - intval($nw);
if ($nk < 0.5) $nw = "" . intval($nw);
else $nw = "" . (intval($nw) + 1);

while (strlen($nw) < 7)
{
    $nw = "0" . $nw;
}

$nk = $ew - intval($ew);
if ($nk < 0.5) $ew = "0" . intval($ew);
else $ew = "0" . intval($ew+1);
return;
}

```

Mit der Funktion utm2geo.php lassen sich UTM Zone, Ost- und Nordwert in geographische Längen und Breiten umrechnen:

```

function utm2geo()
{
    /* Copyright (c) 2006, HELMUT H. HEIMEIER
    Permission is hereby granted, free of charge, to any person obtaining a
    copy of this software and associated documentation files (the "Software"),
    to deal in the Software without restriction, including without limitation
    the rights to use, copy, modify, merge, publish, distribute, sublicense,
    and/or sell copies of the Software, and to permit persons to whom the
    Software is furnished to do so, subject to the following conditions:
    The above copyright notice and this permission notice shall be included
    in all copies or substantial portions of the Software.*/

    /* Die Funktion wandelt UTM Koordinaten in geographische Koordinaten
    um. UTM Zone, Ostwert ew und Nordwert nw müssen gegeben sein.
    Berechnet werden geographische Länge lw und Breite bw im WGS84 Datum.

    global $rw, $hw, $lp, $bp, $lw, $bw, $zone, $ew, $nw, $raster, $ew2, $nw2;

    // Längenzone zone, Ostwert ew und Nordwert nw im WGS84 Datum
    if ($zone == "" || $ew == "" || $nw == "")
    {
        $zone = "";
        $ew = "";
        $nw = "";
        return;
    }
    $band = substr($zone,2,1);
    $z1 = intval($zone);
    $ew1 = intval($ew);
    $nw1 = intval($nw);

    //WGS 84 Datum
    //Große Halbachse a und Abplattung f
    $a = 6378137.000;
    $f = 3.35281068e-3;

    //Polkrümmungshalbmesser c
    $c = $a/(1-$f);

```

```

//Quadrat der zweiten numerischen Exzentrizität
$ex2 = (2*$f-$f*$f)/((1-$f)*(1-$f));
$ex4 = $ex2*$ex2;
$ex6 = $ex4*$ex2;
$ex8 = $ex4*$ex4;

//Koeffizienten zur Berechnung der geographischen Breite aus gegebener
//Meridianbogenlänge
$e0 = $c*(pi()/180)*(1 - 3*$ex2/4 + 45*$ex4/64 - 175*$ex6/256 + 11025*$ex8/16384);
$f2 = (180/pi())*( 3*$ex2/8 - 3*$ex4/16 + 213*$ex6/2048 - 255*$ex8/4096);
$f4 = (180/pi())*( 21*$ex4/256 - 21*$ex6/256 + 533*$ex8/8192);
$f6 = (180/pi())*( 151*$ex6/6144 - 453*$ex8/12288);

//Entscheidung Nord-/Süd Halbkugel
if ($band >= "N" || $band == "")
    $m_nw = $nw1;
else
    $m_nw = $nw1 - 10e6;

//Geographische Breite bf zur Meridianbogenlänge gf = m_nw
$sigma = ($m_nw/0.9996)/$e0;
$sigmr = $sigma*pi()/180;
$bf = $sigma + $f2*sin(2*$sigmr) + $f4*sin(4*$sigmr) + $f6*sin(6*$sigmr);

//Breite bf in Radianen
$br = $bf * pi()/180;
$tan1 = tan($br);
$tan2 = $tan1*$tan1;
$tan4 = $tan2*$tan2;

$cos1 = cos($br);
$cos2 = $cos1*$cos1;

$etasq = $ex2*$cos2;

//Querkrümmungshalbmesser nd
$nd = $c/sqrt(1 + $etasq);
$nd2 = $nd*$nd;
$nd4 = $nd2*$nd2;
$nd6 = $nd4*$nd2;
$nd3 = $nd2*$nd;
$nd5 = $nd4*$nd;

//Längendifferenz dl zum Bezugsmeridian lh
$lh = ($z1 - 30)*6 - 3;
$dy = ($ew1-500000)/0.9996;
$dy2 = $dy*$dy;
$dy4 = $dy2*$dy2;
$dy3 = $dy2*$dy;
$dy5 = $dy2*$dy3;
$dy6 = $dy3*$dy3;

$b2 = - $tan1*(1+$etasq)/(2*$nd2);
$b4 = $tan1*(5+3*$tan2+6*$etasq*(1-$tan2))/(24*$nd4);
$b6 = - $tan1*(61+90*$tan2+45*$tan4)/(720*$nd6);

$l1 = 1/($nd*$cos1);
$l3 = - (1+2*$tan2+$etasq)/(6*$nd3*$cos1);
$l5 = (5+28*$tan2+24*$tan4)/(120*$nd5*$cos1);

//Geographische Breite bw und Länge lw als Funktion von Ostwert ew
// und Nordwert nw
$bw = $bf + (180/pi()) * ($b2*$dy2 + $b4*$dy4 + $b6*$dy6);
$lw = $lh + (180/pi()) * ($l1*$dy + $l3*$dy3 + $l5*$dy5);
return;
}

```

Die beiden folgenden Funktionen rechnen im WGS84 Kartenbezugssystem (world geodetic system 84) zivile UTM Koordinaten in MGRS (military grid reference system), auch UTMREF Koordinaten genannt, um und umgekehrt.

Die Funktion utm2mgr.php ermittelt aus UTM Gitterzone und 7-stelligem Ost- und Nordwert das UTMREF Raster mit 5-stelligem Ost- und Nordwert:

```
function utm2mgr()
{
/* Copyright (c) 2006, HELMUT H. HEIMEIER
Permission is hereby granted, free of charge, to any person obtaining a
copy of this software and associated documentation files (the "Software"),
to deal in the Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense,
and/or sell copies of the Software, and to permit persons to whom the
Software is furnished to do so, subject to the following conditions:
The above copyright notice and this permission notice shall be included
in all copies or substantial portions of the Software.*/

/* Die Funktion wandelt zivile UTM Koordinaten in militärische Koordinaten
* um. UTM Zone zone, Ostwert ew und Nordwert nw müssen gegeben sein.
Zurückgegeben wird das Rasterfeld raster sowie die aus den
letzten 5 Stellen von Ost- und Nordwert gebildete Koordinatenangabe
UTMREF.

global $rw, $hw, $lp, $bp, $lw, $bw, $zone, $ew, $nw, $raster, $ew2, $nw2;

// Längenzone zone, Ostwert ew und Nordwert nw im WGS84 Datum
if ($zone == "" || $ew == "" || $nw == "")
{
$zone = "";
$ew = "";
$nw = "";
return;
}

$z1 = substr($zone,0,2);
$z2 = substr($zone,2,1);
$ew1 = intval(substr($ew,0,2));
$nw1 = intval(substr($nw,0,2));
$ew2 = substr($ew,2,5);
$nw2 = substr($nw,2,5);

$m_east = 'ABCDEFGHJKLMNPQRSTUVWXYZ';
$m_north = 'ABCDEFGHJKLMNPQRSTUV';

if ($z1 < "01" || $z1 > "60" || $z2 < "C" || $z2 > "X")
echo "$zone ist keine gültige UTM Zonenangabe";

$i = $z1 % 3;
if ($i == 1) $m_ce = $ew1 - 1;
if ($i == 2) $m_ce = $ew1 + 7;
if ($i == 0) $m_ce = $ew1 + 15;

$i = $z1 % 2;
if ($i == 1) $m_cn = 0;
else $m_cn = 5;

$i = $nw1;
while ($i-20 >= 0) $i = $i-20;
$m_cn = $m_cn + $i;
if ($m_cn > 19) $m_cn = $m_cn - 20;

$raster = $zone . substr($m_east,$m_ce,1) . substr($m_north,$m_cn,1);
return;
}
```

Die Funktion mgr2utm.php ermittelt aus UTMREF Raster und 5-stelligem Ost- und Nordwert die UTM Gitterzone und 7-stelligen Ost- und Nordwert:

```
function mgr2utm()
{
/* Copyright (c) 2006, HELMUT H. HEIMEIER
Permission is hereby granted, free of charge, to any person obtaining a
copy of this software and associated documentation files (the "Software"),
to deal in the Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense,
and/or sell copies of the Software, and to permit persons to whom the
Software is furnished to do so, subject to the following conditions:
The above copyright notice and this permission notice shall be included
in all copies or substantial portions of the Software.*/

/* Die Funktion wandelt militärische UTM Koordinaten (MGR oder
UTMREF) in zivile UTM Koordinaten um.
UTM Zone zone, raster und utmref müssen gegeben sein.
In zone muss die aus 2 Ziffern bestehende Längenzone enthaltens ein
gefolgt von der aus einem Buchstaben bestehenden Bandangabe.
In raster muss die aus 2 Buchstaben bestehende Kennung für das
100 km x 100 km Rasterfeld enthalten sein.
In UTMREF muss der 5 stellige Ostwert stehen gefolgt von einem blank
und dem 5 stelligen Nordwert.
Berechnet wird daraus der 7 stellige Ost- und Nordwert im zivilen
UTM System.

global $rw, $hw, $lp, $bp, $lw, $bw, $zone, $ew, $nw, $raster, $ew2, $nw2;

// Längenzone zone, Ostwert ew und Nordwert nw im WGS84 Datum
if ($raster == "" || $ew2 == "" || $nw2 == "")
{
$raster = "";
$ew2 = "";
$nw2 = "";
return;
}

$m_east_0 = "STUVWXYZ";
$m_east_1 = "ABCDEFGH";
$m_east_2 = "JKLMNPQR";
$m_north_0 = "FGHJKLMNPQRSTUVWXYZ";
$m_north_1 = "ABCDEFGHJKLMNPQRSTUVWXYZ";

$zone = substr($raster,0,3);
$r_east = substr($raster,3,1);
$r_north = substr($raster,4,1);

$i = intval(substr($raster,0,2)) % 3;
if ($i == 0) $m_ce = strpos($m_east_0,$r_east)+1;
if ($i == 1) $m_ce = strpos($m_east_1,$r_east)+1;
if ($i == 2) $m_ce = strpos($m_east_2,$r_east)+1;
$m_ce = "0" . $m_ce;
$ew = $m_ce . $ew2;

$i = intval(substr($raster,0,2)) % 2;
if ($i == 0) $m_cn = strpos($m_north_0,$r_north);
else $m_cn = strpos($m_north_1,$r_north);

$band = substr($raster,2,1);
if ($band >= "N"){
if ($band == "Q" && $m_cn < 10)
$m_cn = $m_cn + 20;
if ($band >= "R")
$m_cn = $m_cn + 20;
if ($band == "S" && $m_cn < 30)
$m_cn = $m_cn + 20;
if ($band >= "T")
$m_cn = $m_cn + 20;
```

```

    if ($band == "U" && $m_cn < 50)
        $m_cn = $m_cn + 20;
    }
    else {
        if ($band == "C" && $m_cn < 10)
            $m_cn = $m_cn + 20;
        if ($band >= "D")
            $m_cn = $m_cn + 20;
        if ($band == "F" && $m_cn < 30)
            $m_cn = $m_cn + 20;
        if ($band >= "G")
            $m_cn = $m_cn + 20;
        if ($band == "H" && $m_cn < 50)
            $m_cn = $m_cn + 20;
        if ($band >= "J")
            $m_cn = $m_cn + 20;
        if ($band == "K" && $m_cn < 70)
            $m_cn = $m_cn + 20;
        if ($band >= "L")
            $m_cn = $m_cn + 20;
    }

    if (strlen($m_cn) == 1)
        $nw = "0" . $m_cn . $nw2;
    else
        $nw = "" . $m_cn . $nw2;
return;
}

```

Umrechnungen für das Bessel-Ellipsoid

Die beiden folgenden Funktionen rechnen für das Potsdam Kartendatum (Bessel Ellipsoid) geographische Koordinaten in Gauss-Krüger Koordinaten um und umgekehrt.

Mit der Funktion geo2gk.php lassen sich geographische Längen und Breiten für das Potsdam Kartendatum (Bessel-Ellipsoid) in deutsche Gauss-Krüger Koordinaten umrechnen:

```

function geo2gk()
{
    /* Copyright (c) 2006, HELMUT H. HEIMEIER
    Permission is hereby granted, free of charge, to any person obtaining a
    copy of this software and associated documentation files (the "Software"),
    to deal in the Software without restriction, including without limitation
    the rights to use, copy, modify, merge, publish, distribute, sublicense,
    and/or sell copies of the Software, and to permit persons to whom the
    Software is furnished to do so, subject to the following conditions:
    The above copyright notice and this permission notice shall be included
    in all copies or substantial portions of the Software.*/

    /* Die Funktion wandelt geographische Koordinaten in GK Koordinaten
    um. Geographische Länge lp und Breite bp müssen im Potsdam Datum
    gegeben sein. Berechnet werden Rechtswert rw und Hochwert hw.

    global $rw, $hw, $lp, $bp, $lw, $bw, $zone, $ew, $nw, $raster, $ew2, $nw2;

    //Geographische Länge lp und Breite bp im Potsdam Datum
    if ($lp == "" || $bp == "")
    {
        $rw = "";
        $hw = "";
        return;
    }
    $lp = doubleval($lp);
    $bp = doubleval($bp);

    // Grenzen des Gauss-Krüger-Systems für Deutschland 46° N < bp < 55° N,
    //                                                    5° E < lp < 16° E
    if ($bp < 46 || $bp > 56 || $lp < 5 || $lp > 16)
    {

```

```
// echo("Werte außerhalb des für Deutschland definierten Gauss-Krüger-Systems\n"+
// " 5° E < LP < 16° E, 46° N < BP < 55° N");
$rw = " ";
$hw = " ";
return;
}

// Potsdam Datum
// Große Halbachse a und Abplattung f
$a = 6377397.155;
$f = 3.34277321e-3;

// Polkrümmungshalbmesser c
$c = $a/(1-$f);

// Quadrat der zweiten numerischen Exzentrizität
$ex2 = (2*$f-$f*$f)/((1-$f)*(1-$f));
$ex4 = $ex2*$ex2;
$ex6 = $ex4*$ex2;
$ex8 = $ex4*$ex4;

// Koeffizienten zur Berechnung der Meridianbogenlänge
$e0 = $c*(pi()/180)*(1 - 3*$ex2/4 + 45*$ex4/64 - 175*$ex6/256 + 11025*$ex8/16384);
$e2 = $c*( - 3*$ex2/8 + 15*$ex4/32 - 525*$ex6/1024 + 2205*$ex8/4096);
$e4 = $c*(15*$ex4/256 - 105*$ex6/1024 + 2205*$ex8/16384);
$e6 = $c*( - 35*$ex6/3072 + 315*$ex8/12288);

// Breite in Radianen
$br = $bp * pi()/180;

$tan1 = tan($br);
$tan2 = $tan1*$tan1;
$tan4 = $tan2*$tan2;

$cos1 = cos($br);
$cos2 = $cos1*$cos1;
$cos4 = $cos2*$cos2;
$cos3 = $cos2*$cos1;
$cos5 = $cos4*$cos1;

$etasq = $ex2*$cos2;

// Querkrümmungshalbmesser nd
$nd = $c/sqrt(1 + $etasq);

// Meridianbogenlänge g aus gegebener geographischer Breite bp
$g = $e0*$bp + $e2*sin(2*$br) + $e4*sin(4*$br) + $e6*sin(6*$br);

// Längendifferenz dl zum Bezugsmeridian lh
$kz = intval(($lp+1.5)/3);
$lh = $kz*3;
$d1 = ($lp - $lh)*pi()/180;
$d12 = $d1*$d1;
$d14 = $d12*$d12;
$d13 = $d12*$d1;
$d15 = $d14*$d1;

// Hochwert hw und Rechtswert rw als Funktion von geographischer Breite und Länge
$hw = ($g + $nd*$cos2*$tan1*$d12/2 + $nd*$cos4*$tan1*(5-$tan2+9*$etasq)
      *$d14/24);
$rw = ($nd*$cos1*$d1 + $nd*$cos3*(1-$tan2+$etasq)*$d13/6 +
      $nd*$cos5*(5-18*$tan2+$tan4)*$d15/120 + $kz*1e6 + 500000);

$nk = $hw - intval($hw);
if ($nk < 0.5) $hw = intval($hw);
else $hw = intval($hw) + 1;

$nk = $rw - intval($rw);
if ($nk < 0.5) $rw = intval($rw);
```



```

    else $rw = intval($rw+1);
    return;
}

```

Mit der Funktion gk2geo.php lassen sich Gauss-Krüger Koordinaten in geographische Längen und Breiten des Potsdam Datums umrechnen:

```

function gk2geo()
{
/* Copyright (c) 2007, HELMUT H. HEIMEIER
Permission is hereby granted, free of charge, to any person obtaining a
copy of this software and associated documentation files (the "Software"),
to deal in the Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense,
and/or sell copies of the Software, and to permit persons to whom the
Software is furnished to do so, subject to the following conditions:
The above copyright notice and this permission notice shall be included
in all copies or substantial portions of the Software.*/

/* Die Funktion wandelt GK Koordinaten in geographische Koordinaten
um. Rechtswert rw und Hochwert hw müssen gegeben sein.
Berechnet werden geographische Länge lp und Breite bp
im Potsdam Datum.

global $rw, $hw, $lp, $bp, $lw, $bw, $zone, $ew, $nw, $raster, $ew2, $nw2;

// Rechtswert rw und Hochwert hw im Potsdam Datum
if ($rw == "" || $hw == "")
{
$lp = "";
$bp = "";
return;
}

$rw = intval($rw);
$hw = intval($hw);

// Potsdam Datum
// Große Halbachse a und Abplattung f
$a = 6377397.155;
$f = 3.34277321e-3;
// Polkrümmungshalbmesser c
$c = $a/(1-$f);

// Quadrat der zweiten numerischen Exzentrizität
$ex2 = (2*$f-$f*$f)/((1-$f)*(1-$f));
$ex4 = $ex2*$ex2;
$ex6 = $ex4*$ex2;
$ex8 = $ex4*$ex4;

// Koeffizienten zur Berechnung der geographischen Breite aus gegebener
// Meridianbogenlänge
$e0 = $c*(pi()/180)*(1 - 3*$ex2/4 + 45*$ex4/64 - 175*$ex6/256 + 11025*$ex8/16384);
$f2 = (180/pi())*( 3*$ex2/8 - 3*$ex4/16 + 213*$ex6/2048 - 255*$ex8/4096);
$f4 = (180/pi())*( 21*$ex4/256 - 21*$ex6/256 + 533*$ex8/8192);
$f6 = (180/pi())*( 151*$ex6/6144 - 453*$ex8/12288);

// Geographische Breite bf zur Meridianbogenlänge gf = hw
$sigma = $hw/$e0;
$sigmr = $sigma*pi()/180;
$bf = $sigma + $f2*sin(2*$sigmr) + $f4*sin(4*$sigmr) + $f6*sin(6*$sigmr);

// Breite bf in Radianten
$br = $bf * pi()/180;
$tan1 = tan($br);
$tan2 = $tan1*$tan1;
$tan4 = $tan2*$tan2;

$cos1 = cos($br);

```

```

$cos2 = $cos1*$cos1;

$etasq = $ex2*$cos2;

// Querkrümmungshalbmesser nd
$nd = $c/sqrt(1 + $etasq);
$nd2 = $nd*$nd;
$nd4 = $nd2*$nd2;
$nd6 = $nd4*$nd2;
$nd3 = $nd2*$nd;
$nd5 = $nd4*$nd;

// Längendifferenz dl zum Bezugsmeridian lh
$kz = intval($rw/1e6);
$lh = $kz*3;
$dy = $rw-($kz*1e6+500000);
$dy2 = $dy*$dy;
$dy4 = $dy2*$dy2;
$dy3 = $dy2*$dy;
$dy5 = $dy4*$dy;
$dy6 = $dy3*$dy3;

$b2 = - $tan1*(1+$etasq)/(2*$nd2);
$b4 =  $tan1*(5+3*$tan2+6*$etasq*(1-$tan2))/(24*$nd4);
$b6 = - $tan1*(61+90*$tan2+45*$tan4)/(720*$nd6);

$l1 =  1/($nd*$cos1);
$l3 = - (1+2*$tan2+$etasq)/(6*$nd3*$cos1);
$l5 =  (5+28*$tan2+24*$tan4)/(120*$nd5*$cos1);

// Geographischer Breite bp und Länge lp als Funktion von Rechts- und Hochwert
$bp = $bf + (180/pi()) * ($b2*$dy2 + $b4*$dy4 + $b6*$dy6);
$lp = $lh + (180/pi()) * ($l1*$dy + $l3*$dy3 + $l5*$dy5);

if ($lp < 5 || $lp > 16 || $bp < 46 || $bp > 56)
{
    echo("RW und/oder HW ungültig für das deutsche Gauss-Krüger-System
\n");
    $lp = "";
    $bp = "";
}
return;
}

```

Funktionen zur Änderung des Kartenbezugssystems WGS84 - Potsdam

Mit den beiden folgenden Funktionen lässt sich das Kartenbezugsdatum vom WGS84 System auf das Potsdam Datum (Zentralpunkt Rauenberg) und umgekehrt verschieben. Bei der Transformation werden die Ellipsoidachsen parallel verschoben um $dx = 587$ m, $dy = 16$ m und $dz = 393$ m. Hierbei liegt der Koordinatenursprung im Erdmittelpunkt, die positive x-Achse schneidet Äquator und Nullmeridian, die positive y-Achse schneidet Äquator und 90 grad E Meridian und die positive z-Achse schneidet den Nordpol. Die shift-Parameter dx, dy und dz sind so gewählt, dass sich bei der Umformung Übereinstimmung mit den von Garmin GPS-Geräten ermittelten Werten ergibt.

Man beachte, dass in der deutschen Landesvermessung im allgemeinen mit anderen Parametern gerechnet wird, nämlich mit $dx = 631$ m, $dy = 23$ m und $dz = 451$ m.

Die Funktion wgs2pot.php formt geographischen Längen und Breiten des WGS84 Datums in solche des Potsdam Datums um:

```

function wgs2pot()
{
/* Copyright (c) 2006, HELMUT H. HEIMEIER
Permission is hereby granted, free of charge, to any person obtaining a
copy of this software and associated documentation files (the "Software"),
to deal in the Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense,
and/or sell copies of the Software, and to permit persons to whom the

```

Software is furnished to do so, subject to the following conditions:
 The above copyright notice and this permission notice shall be included
 in all copies or substantial portions of the Software.*/

```
/* Die Funktion verschiebt das Kartenbezugssystem (map datum) vom
WGS84 Datum (World Geodetic System 84) zum in Deutschland
gebräuchlichen Potsdam-Datum. Geographische Länge lw und Breite
bw gemessen in grad auf dem WGS84 Ellipsoid müssen
gegeben sein. Ausgegeben werden geographische Länge lp
und Breite bp (in grad) auf dem Bessel-Ellipsoid.
Bei der Transformation werden die Ellipsoidachsen parallel
verschoben um dx = -587 m, dy = -16 m und dz = -393 m.

global $rw, $hw, $lp, $bp, $lw, $bw, $zone, $ew, $nw, $raster, $ew2, $nw2;

// Geographische Länge lw und Breite bw im WGS84 Datum
if ($lw == "" || $bw == "")
{
  $lp = "";
  $bp = "";
  return;
}
$lw = doubleval($lw);
$bw = doubleval($bw);

// Quellsystem WGS 84 Datum
// Große Halbachse a und Abplattung fq
$a = 6378137.000;
$fq = 3.35281066e-3;

// Zielsystem Potsdam Datum
// Abplattung f
$f = $fq - 1.003748e-5;

// Parameter für datum shift
$dx = -587;
$dy = -16;
$dz = -393;

// Quadrat der ersten numerischen Exzentrizität in Quell- und Zielsystem
$e2q = (2*$fq-$fq*$fq);
$e2 = (2*$f-$f*$f);

// Breite und Länge in Radianen
$b1 = $bw * (pi()/180);
$l1 = $lw * (pi()/180);

// Querkrümmungshalbmesser nd
$nd = $a/sqrt(1 - $e2q*sin($b1)*sin($b1));

// Kartesische Koordinaten des Quellsystems WGS84
$xw = $nd*cos($b1)*cos($l1);
$yw = $nd*cos($b1)*sin($l1);
$zw = (1 - $e2q)*$nd*sin($b1);

// Kartesische Koordinaten des Zielsystems (datum shift) Potsdam
$x = $xw + $dx;
$y = $yw + $dy;
$z = $zw + $dz;

// Berechnung von Breite und Länge im Zielsystem
$rb = sqrt($x*$x + $y*$y);
$b2 = (180/pi()) * atan(($z/$rb)/(1-$e2));

if ($x > 0)
  $l2 = (180/pi()) * atan($y/$x);
if ($x < 0 && $y > 0)
  $l2 = (180/pi()) * atan($y/$x) + 180;
if ($x < 0 && $y < 0)
```

```

    $l2 = (180/pi()) * atan($y/$x) - 180;

    $lp = $l2;
    $bp = $b2;

/*    if ($lp < 5 || $lp > 16 || $bp < 46 || $bp > 56)
    {
        $lp = "";
        $bp = "";
    }*/
    return;
}

```

Die Funktion pot2wgs.php formt geographische Längen und Breiten des Potsdam Datums in solche des WGS84 Datums um:

```

function pot2wgs()
{
/* Copyright (c) 2006, HELMUT H. HEIMEIER
   Permission is hereby granted, free of charge, to any person obtaining a
   copy of this software and associated documentation files (the "Software"),
   to deal in the Software without restriction, including without limitation
   the rights to use, copy, modify, merge, publish, distribute, sublicense,
   and/or sell copies of the Software, and to permit persons to whom the
   Software is furnished to do so, subject to the following conditions:
   The above copyright notice and this permission notice shall be included
   in all copies or substantial portions of the Software.*/

/* Die Funktion verschiebt das Kartenbezugssystem (map datum) vom in
   Deutschland gebräuchlichen Potsdam-Datum zum WGS84 (World Geodetic
   System 84) Datum. Geographische Länge lp und Breite bp gemessen in
   grad auf dem Bessel-Ellipsoid müssen gegeben sein.
   Ausgegeben werden geographische Länge lw und
   Breite bw (in grad) auf dem WGS84-Ellipsoid.
   Bei der Transformation werden die Ellipsoidachsen parallel
   verschoben um dx = 587 m, dy = 16 m und dz = 393 m.

   global $rw, $hw, $lp, $bp, $lw, $bw, $zone, $ew, $nw, $raster, $ew2, $nw2;

// Geographische Länge lp und Breite bp im Potsdam Datum
   if ($lp == "" || $bp == "")
   {
       $lw = "";
       $bw = "";
       return;
   }

   $lp = doubleval($lp);
   $bp = doubleval($bp);

// Quellsystem Potsdam Datum
// Große Halbachse a und Abplattung fq
   $a = 6378137.000 - 739.845;
   $fq = 3.35281066e-3 - 1.003748e-05;

// Zielsystem WGS84 Datum
// Abplattung f
   $f = 3.35281066e-3;

// Parameter für datum shift
   $dx = 587;
   $dy = 16;
   $dz = 393;

// Quadrat der ersten numerischen Exzentrizität in Quell- und Zielsystem
   $e2q = (2*$fq-$fq*$fq);
   $e2 = (2*$f-$f*$f);

// Breite und Länge in Radianen

```

```
$b1 = $bp * (pi()/180);
$l1 = $lp * (pi()/180);

// Querkrümmungshalbmesser nd
$nd = $a/sqrt(1 - $e2q*sin($b1)*sin($b1));

// Kartesische Koordinaten des Quellsystems Potsdam
$x_p = $nd*cos($b1)*cos($l1);
$y_p = $nd*cos($b1)*sin($l1);
$z_p = (1 - $e2q)*$nd*sin($b1);

// Kartesische Koordinaten des Zielsystems (datum shift) WGS84
$x = $x_p + $dx;
$y = $y_p + $dy;
$z = $z_p + $dz;

// Berechnung von Breite und Länge im Zielsystem
$rb = sqrt($x*$x + $y*$y);
$b2 = (180/pi()) * atan(($z/$rb)/(1-$e2));

if ($x > 0)
    $l2 = (180/pi()) * atan($y/$x);
if ($x < 0 && $y > 0)
    $l2 = (180/pi()) * atan($y/$x) + 180;
if ($x < 0 && $y < 0)
    $l2 = (180/pi()) * atan($y/$x) - 180;

$lw = $l2;
$bw = $b2;
return;
}
```