Arbeitstage in einem Datumsbereich ermitteln

Dr. V. Thormählen, 7. Juni 2019

Inhalt

Abbildungen

Listings

Tabellen

1 Aufgabe

In diesem Beitrag wird beschrieben, wie die Zahl der Arbeitstage in einem auswählten Datumsbereich und einem auswählten deutschen Bundesland mit PHP berechnet werden kann, wenn …

- eine 5-Tage-Woche zugrunde gelegt wird und
- alle Wochenfeiertage des jeweils gewählten deutschen Bundeslands berücksichtigt werden.

Im Folgenden wird zunächst das Eingabeformular vorgestellt, zuerst belegt mit fehlerhaften Eingabewerten (s. Abbildung 1) und danach mit gültigen Eingabewerten (s. Abbildung 2).

Danach werden fünf PHP-Funktionen zur Ermittlung der Zahl der Arbeitstage je nach gewähltem deutschen Bundesland und gewähltem Datumsbereich vorgestellt (s. Tabelle 1).

Name der PHP-Funktion	Aufgabe
date_picker	Datumsbereich im Eingabeformular interaktiv auswählen.
loop_thru_date_range	Ausgewählten Datumsbereich mit einer Zählschleife durchlaufen.
is_week_holiday	Wochenfeiertag nach Datum und deutschem Bundesland ermitteln.
get_week_day_name	Wochentagname eines Datums ermitteln.
test_input	Formularwerte bereinigen.

Tabelle 1: Benutzerdefinierte PHP-Funktionen

2 Eingabeformular

Das Eingabeformular umfasst zwei Feldgruppen: *Bundesland* und *Datumsbereich*. Sie enthalten jeweils 1 oder mehr Auswahllisten zur Einzelauswahl (s. Abbildung 1 und Abbildung 2).

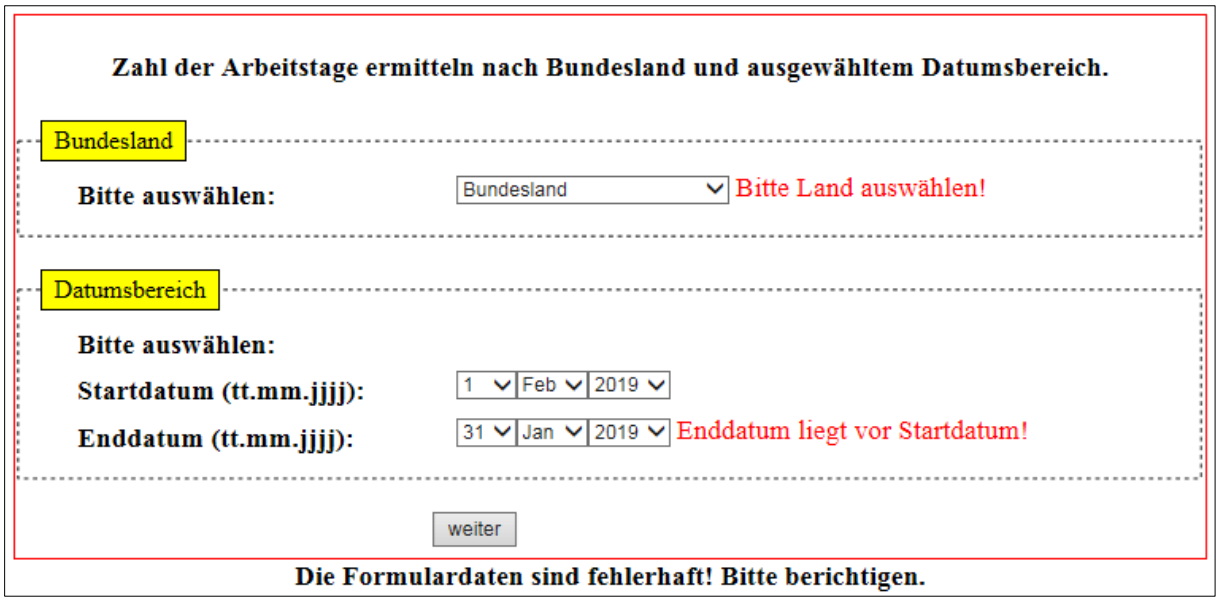

Abbildung 1: Eingabeformular mit fehlerhaften Formularwerten

Fehlermeldungen erscheinen ggf. rot rechts neben dem entsprechenden Eingabefeld (s. Abbildung 1).

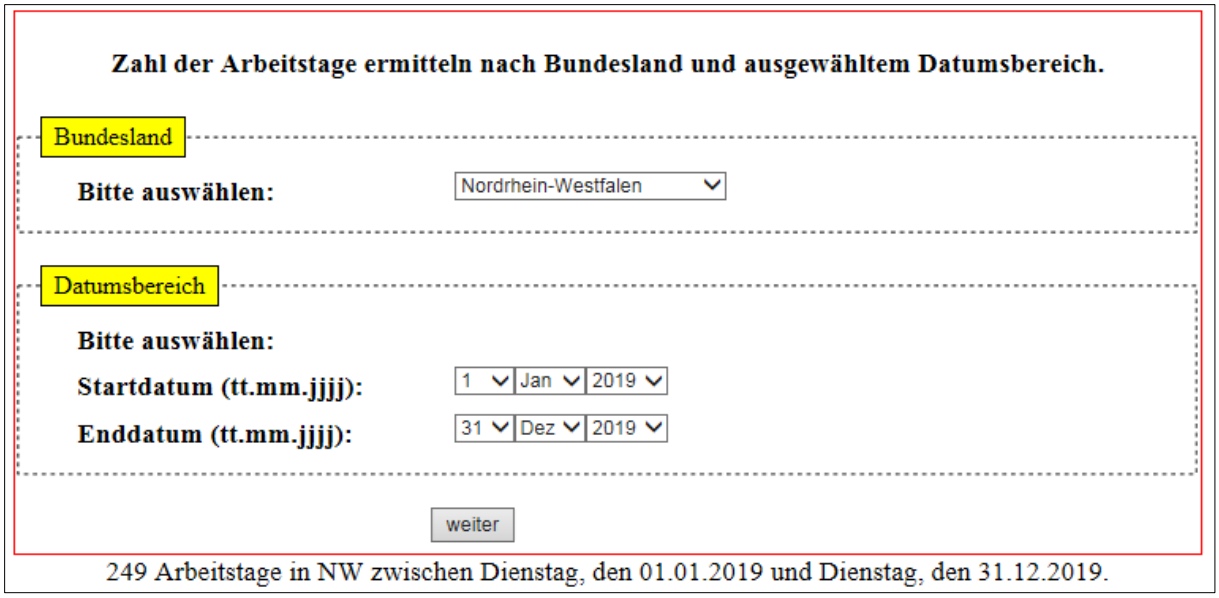

Abbildung 2: Eingabeformular mit gültigen Eingabewerten

Die Besonderheit dieses Formular besteht darin, das alle Eingabewerte nach dem Absenden erhalten bleiben, so dass die Korrektur fehlerhafter Eingaben bequem erfolgen kann.

Um redundanten PHP-Code zu vermeiden wird die Auswahl des Start- und Enddatums in der Feldgruppe '*Datumsbereich*' des Eingabeformulars mit der PHP-Funktion '***date_picker***' durchgeführt (s. Tabelle 1 und Listing 1):

```
function date_picker($tagNm,$startYr,$endYr,$thisDay,$thisMonth) {
//     Aufgabe: Datumsbereich interaktiv auswählen.
   $months=array(' ','Jan','Feb','Mrz','Apr','Mai','Jun','Jul','Aug','Spt','Okt','Nov','Dez');
// Tag
   $html="<select name=\"".$tagNm."day\">";
   for($i=1;$i<=31;$i++){
      $selected = ($i==$thisDay)?' selected="selected"':'';
      $html.="<option ".$selected." value='$i'>$i</option>";
   }
   $html.="</select>";
// Monat
   $html.="<select name=\"".$tagNm."month\">";
   for($i=1;$i<=12;$i++){
      $selected = ($i==$thisMonth)?' selected="selected"':'';
      $html.="<option ".$selected." value='$i'>$months[$i]</option>";
   }
   $html.="</select>";
// Jahr
   $html.="<select name=\"".$tagNm."year\">";
   for($i=$startYr;$i<=$endYr;$i++){
      $html.="<option value='$i'>$i</option>";
   }
   $html.="</select>";
   return $html;
}
```

Listing 1: PHP-Funktion '*date_picker*'

Nicht im Detail gezeigt wird die Formatierung des HTML-Formulars mit CSS-Anweisungen. Verwiesen wird stattdessen auf [1]. Dort sind relevante Details zu finden.

```
<!DOCTYPE html>
<html lang="de">
<head>
   <meta charset="utf-8">
   <title>Arbeitstage nach Bundesland und ausgewähltem Datumsbereich</title>
   <style>
      <!-- CSS-Anweisungen -->
   </style>
</head>
<body>
<?php
// 1. PHP-Datei mit fünf PHP-Funktionen einbinden (s. Tabelle 1)
include_once "Date_Time_Fct.php";)
// 2. Datum und Zeit auf deutsch einstellen.
setlocale(LC_TIME,'de_DE.utf8');
date_default_timezone_set('Europe/Berlin');
// 3. Variablen definieren und Anfangswerte setzen.
$pick1day=$pick1month=$pick1year=$pick2day=$pick2month=$pick2year=$land="";
$pick1_Err=$pick2_Err=$land_Err = "";
$fehlerfrei= true;
$tagNm     = "pick";
$startYr   = $endYr = (int)date('Y');
$thisDay   = date('d');
$thisMonth = date('m');
// 4. Formularwerte prüfen.
if('POST' == $_SERVER['REQUEST_METHOD']) {
   // 4.1 Land auf Gültigkeit prüfen.
   if(empty($_POST['land'])) {
      $land_Err ="Bitte Land auswählen!";
      $fehlerfrei = false;
   } else {
      $land_sel = test_input($_POST['land']);
      if ($land_sel== "Bundesland") {
         $land_Err ="Bitte Bundesland auswählen!";
         $fehlerfrei = false;
      }
   }
   // 4.2 Startdatum auf Gültigkeit prüfen.
   $pick1day  = intval(test_input($_POST["pick1day"]));
   $pick1month = intval(test_input($_POST["pick1month"]));
   $pick1year   = intval(test_input($_POST["pick1year"]));
   if (!checkdate($pick1month,$pick1day,$pick1year)){
      $pick1_Err ="Ungültiges Startdatum!";
      $fehlerfrei = false;
   } else {
      $pick1date = date('Y-m-d', mktime(0,0,0,$pick1month,$pick1day,$pick1year));
   }
   // 4.3. Enddatum auf Gültigkeit prüfen.
   $pick2day  = intval(test_input($_POST["pick2day"]));
   $pick2month = intval(test_input($_POST["pick2month"]));
```

```
   $pick2year   = intval(test_input($_POST["pick2year"]));
   if (!checkdate($pick2month,$pick2day,$pick2year)){
      $pick2_Err ="Ungültiges Enddatum!";
      $fehlerfrei = false;
   } else {
      $pick2date = date("Y-m-d", mktime(0,0,0,$pick2month,$pick2day,$pick2year));
   }
   // 4.4. Plausibilität prüfen: Das Enddatum darf nicht vor dem Startdatum liegen.
   if (!empty($pick1date) && !empty($pick2date) && ($pick1date > $pick2date)) {
      $pick2_Err ="Enddatum liegt vor Startdatum!";
      $fehlerfrei = false;
   }
}
?>
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
   <center><h4>Zahl der Arbeitstage ermitteln nach Bundesland und ausgewähltem
Datumsbereich.</h4></center>
   <fieldset>
      <legend>Bundesland</legend>
      <label for "land">Bitte auswählen:</label>
      <select name="land" size="1">
      <option disabled selected>Bundesland</option>
      <?php
         $land_sel = isset($land_sel) ? $land_sel : "";
         $bundeslaender=array(
         "BW"=>"Baden-Würtemberg","BY"=>"Bayern","BE"=>"Berlin","BB"=>"Brandenburg",
         "HB"=>"Bremen","HH"=>"Hamburg","HE"=>"Hessen","MV"=>"Mecklenburg-
Vorpommern",
         "NI"=>"Niedersachsen","NW"=>"Nordrhein-Westfalen","RP"=>"Rheinland-
Pfalz","SL"=>"Saarland",
         "SN"=>"Sachsen","ST"=>"Sachsen-Anhalt","SH"=>"Schleswig-
Holstein","TH"=>"Thüringen");
         foreach($bundeslaender as $key=>$land){
            $sel = ($land_sel == $key) ? " selected='selected'" : "";
            echo "<option value='".($key)."' {$sel}>{$land}</option>";
         }
      ?>
      </select>
      <span class="error"><?php echo $land_Err;?></span>
   </fieldset><br>
   <fieldset>
      <legend>Datumsbereich</legend>
      <label>Bitte auswählen:</label><br>
         <label>Startdatum (tt.mm.jjjj):</label>
         <?php
            $yr     = !empty($pick1year) ? $pick1year  : $startYr;
            $pickday   = !empty($pick1day)   ? $pick1day   : $thisDay;
            $pickmonth   = !empty($pick1month)? $pick1month : $thisMonth;
            echo date_picker("pick1",$yr,$yr+1,$pickday,$pickmonth);
         ?>
         <span class="error"><?php echo $pick1_Err;?></span>
         <br>
         <label>Enddatum (tt.mm.jjjj):</label>
```

```
        <?php
           $yr     = !empty($pick2year) ? $pick2year  : $endYr;
           $pickday  = !empty($pick2day)   ? $pick2day   : $thisDay;
           $pickmonth   = !empty($pick2month)? $pick2month : $thisMonth;
           echo date_picker("pick2",$yr,$yr+1,$pickday,$pickmonth);
        ?>
        <span class="error"><?php echo $pick2_Err;?></span>
        <br>
   </fieldset><br>
   <label for="submit"> </label>
   <input type="submit" name="submit" value="weiter" />
   <br>
</form>
</body>
</html>
```

Listing 2: Eingabeformular

In Pkt. 4 (*Formularwerte prüfen*) von Listing 2 wird die PHP-Funktion **'test_input'** mehrmals aufgerufen, um die erfassten Formularwerte nach dem Absenden des Formulars zu validieren (s. Listing 3).

```
function test_input($data) {
   // Aufgabe: Formularwerte prüfen und ggf. bereinigen.
   $data = trim($data);       // Überflüssige Zeichen am Anfang und Ende entfernen.
   $data = stripslashes($data);       // Maskierungszeichen entfernen.
   $data = htmlspecialchars($data);   // Sonderzeichen in HTML-Codes umwandeln.
   return $data;
}
```

Listing 3: Formularwerte prüfen und ggf. bereinigen

3 Auswertung des Eingabeformulars

```
…
<form>
<!-- Eingabeformular, s. Listing 2 -->
</form>
</body>
</html>
<?php
if(isset($_POST['submit'])){
// 5. Status der Formularwerte prüfen.
   if (!$fehlerfrei) { echo "<center><b>Die Formulardaten sind fehlerhaft! Bitte
berichtigen.</b></center>";
      exit();
   }
// 6. Bundesland aufbereiten.
   $land_sel = $_POST['land'];
// 7. Startdatum aufbereiten.
   $month1 = $_POST['pick1month'];
   $day1   = $_POST['pick1day'];
   $year1  = $_POST['pick1year'];
   if(strlen($day1) < 2) {
      $day1 = str_pad($day1,2,"0",STR_PAD_LEFT);
```

```
   }
   if(strlen($month1) < 2) {
      $month1 = str_pad($month1,2,"0",STR_PAD_LEFT);
   }
   $startDt= $year1."-".$month1."-".$day1;
// 8. Enddatum aufbereiten.
   $month2 = $_POST['pick2month'];
   $day2   = $_POST['pick2day'];
   $year2  = $_POST['pick2year'];
   if(strlen($day2) < 2) {
      $day2 = str_pad($day2,2,"0",STR_PAD_LEFT);
   }
   if(strlen($month2) < 2) {
      $month2 = str_pad($month2,2,"0",STR_PAD_LEFT);
   }
   $endDt= $year2."-".$month2."-".$day2;
// 9. Datumsbereich durchlaufen, um die Arbeitstage ($workDays)
// im ausgewählten Datumsbereich und Bundesland zu ermitteln.
   $workDays = loop_thru_date_range($startDt,$endDt,$land_sel);
// 10. Wochentagsname des Endtags bestimmen.
   $endTgNm = get_week_day_name($endDt);
// 11. Wochentagsname des Starttags bestimmen.
   $startTgNm=get_week_day_name($startDt);
// 12. Arbeittage im ausgewählten Bundesland und ausgewähltem Datumsbereich
ausgeben.
   ?><center><?php
      echo $workDays." Arbeitstage in ".$land_sel." zwischen ".$startTgNm." und
".$endTgNm.".";
   ?></center><?php
}
?>
```

Listing 4: Auswertung des Eingabeformulars

4 Ausgewählten Datumsbereich mit einer Zählschleife durchlaufen

Die benutzerdefinierte PHP-Funktion mit dem Namen ***'loop_thru_date_range'*** *besitzt die Aufgabe, einen vom* Benutzer im Eingabeformular ausgewähltem Datumsbereich mit einer Zählschleife zu durchlaufen. Sie erfüllt dabei die 6 grau hinterlegte Teilaufgaben (s. Listing 5):

```
function loop_thru_date_range($startDate,$endDate,$land) {
   // Aufgabe: Ausgewählten Datumsbereich mit einer Zählschleife durchlaufen.
   // 1. Abstand in Kalendertagen.
   $datediff = (strtoTime($endDate) - strtoTime($startDate));
   $days = intval(floor($datediff/86400)+1);
   // 2. Anfangswerte setzen für …
   $count = 0;             // Arbeitstageszähler.
   $currDate = $startDate; // aktuelles Datum.
   // 3. Alle Kalendertage des Datumsbereichs durchlaufen.
   for($day=1;$day<=$days;$day++) { // Zählschleife.
      $weekDay = date('w',strtotime($currDate));
      if($weekDay > 0 && $weekDay < 6 && is_week_holiday($currDate,$land) == "Arbeitstag") {
         // 4. Arbeitstage kumulieren.
         $count ++;
      }
      // 5. Tagesdatum um 1 Kalendertag erhöhen.
      $currDate= date("Y-m-d",strtotime("+1 day",strtotime($currDate)));
   }
   // 6. Gezählte Arbeitstage zurückgeben.
   return $count;
}
```

Listing 5: Einen ausgewählten Datumsbereich mit einer Zählschleife durchlaufen.

5 Wochenfeiertag nach Datum und Bundesland ermitteln

Der zugehörige Quellcode befindet sich in Listing 6.

```
function is_week_holiday ($datum,$land=null) {
   // Aufgabe: Wochenfeiertag nach Datum und Bundesland ermitteln.
   // 1. Prüfen,ob das übergebene Datum skalar ist.
   if(!is_scalar($datum)) {
      echo "Datum ist nicht skalar";
      return false;
   }
   // 2. Datum prüfen.
   if(count(explode('-', $datum)) != 3){
      echo "Datum ist nicht gültig";
      return false;
   } else {
      // Übergebenes Datum aufteilen in Jahr, Monat und Tag.
      list($jahr,$monat,$tag) = explode('-',$datum);
   }
   // 3. Das Jahr muss 4-stellig sein.
   if(!strlen($jahr) == 4) {
      echo "Jahr ist nicht 4-stellig";
      return false;
   }
   // 4. Monat bzw. Tag ggf. linksseitig mit '0' auffüllen.
   if(strlen($monat) < 2) {
      $monat = str_pad($monat,2,"0",STR_PAD_LEFT);
```

```
   }
   if(strlen($tag) < 2) {
      $tag = str_pad($tag,2,"0",STR_PAD_LEFT);
   }
   // 5. Übergebenes Datum mit der PHP-Funktion 'checkdate' auf Gültigkeit prüfen.
   if(!checkdate(intval($monat),intval($tag),intval($jahr))) {
      echo "Übergebenes Datum ist falsch";
      return false;
   }
   // 6. Ostersonntag und -monat bestimmen.
   $ostern  = easter_date($jahr);
   $osterSo    = date("d",$ostern); // Ostertag
   $osterMnth  = date("m",$ostern); // Ostermonat
   // 7. Datums- und Zeitinformationen mit der PHP-Funktion 'getdate' ermitteln.
   $datum_arr = getdate(mktime(0,0,0,$monat,$tag,$jahr));
   // 8. Tag der Woche und zugehörigen Status bestimmen.
   $status =
   ($datum_arr['wday']==0||$datum_arr['wday']==6)?'Wochenende':'Arbeitstag';
   // 6. Monat und Tag verketten und prüfen.
   $md = $monat.$tag;
   if (strlen($md) != 4) {
      echo "Konkatenierung Monat.Tag misslungen";
      return false;
   }
   // 7. Ermitteln, ob das verkettete Datum auf einen Wochenfeiertag fällt.
   if ($md=='0101') {
      return 'Neujahr';
   } elseif ($md=='0106' && ($land=='BW'||$land=='BY'||$land=='ST')){
      return 'Heilige Drei Könige';
   } elseif ($md=='0308' && $land == "BE") {
      return 'Frauentag';
   } elseif ($md==date("md",mktime(0,0,0,$osterMnth,$osterSo-2,$jahr))){
      return 'Karfreitag';
   } elseif ($md==date("md",mktime(0,0,0,$osterMnth,$osterSo+1,$jahr))){
      return 'Ostermontag';
   } elseif ($md=='0501') {
      return 'Tag der Arbeit';
   } elseif ($md==date("md",mktime(0,0,0,$osterMnth,$osterSo+39,$jahr))){
      return 'Christi Himmelfahrt';
   } elseif ($md==date("md",mktime(0,0,0,$osterMnth,$osterSo+50,$jahr))){
      return 'Pfingstmontag';
   } elseif ($md==date("md",mktime(0,0,0,$osterMnth,$osterSo+60,$jahr))
      && ($land=='BW'||$land=='BY'||$land=='HE'||$land=='NW'||$land=='RP'||$land=='SL')){
      return 'Fronleichnam';
   } elseif ($md=='0808' && $land=='BY'){
      return 'Augsburger Friedensfest';
   } elseif ($md=='0920' && $land=='TH'){
      return 'Weltkindertag';
   } elseif ($md=='0815' && ($land=='BY'||$land=='SL')){
      return 'Mariä Himmelfahrt';
   } elseif ($md=='1003') {
      return 'Tag der deutschen Einheit';
   } elseif ($md=='1031' && ($land=='BB'||$land='HB'||$land='HH'||$land=='MV'||$land=='NI'
                  ||$land=='SH'||$land=='SN'||$land=='SH'||$land=='TH')){
      return 'Reformationstag';
   } elseif ($md=='1101' &&
($land=='BW'||$land=='BY'||$land=='NW'||$land=='RP'||$land=='SL')){
      return 'Allerheiligen';
   } elseif ($md==date('md',(strtotime("-11 days",strtotime("1 sunday",
         mktime(0,0,0,11,26,$jahr))))) && $land=='SN') {
```

```
      return 'Buß- und Bettag';
   } elseif ($md=='1225') {
      return '1. Weihnachtstag';
   } elseif ($md=='1226') {
      return '2. Weihnachtstag';
   } else {
      return $status;
   }
}
```

Listing 6: Wochenfeiertag ermitteln und zurückgeben

Die PHP-Funktion '***is_week_holiday***' verwendet einen gestaffelten *If-Then-Else-Block*, um mehrere Anweisungssequenzen zu definieren, von denen jeweils nur eine ausgeführt wird (s. Listing 6).

Im Bundesland Nordrhein-Westfalen (NW) gibt es im Jahr 2019 elf Wochenfeiertage (s. Tabelle 2):

Lfd. Nr.	Wochenfeiertag	tt.mm	Tag	Typ[1]	Berechnung
1	Neujahr	01.01.	Di.	fest	keine
2	Karfreitag	19.04.	Fr.	*beweglich*	Ostern – 2 Kalendertage
3	Ostermontag	22.04.	Mo.	*beweglich*	Ostern + 1 Kalendertage
4	Tag der Arbeit	01.05.	Mi.	fest	keine
5	Himmelfahrt	30.05.	Do.	*beweglich*	Ostern + 39 Kalendertage
6	Pfingstmontag	10.06.	Mo.	*beweglich*	Ostern + 50 Kalendertage
7	Fronleichnam	20.06.	Do.	*beweglich*	Ostern + 60 Kalendertage
8	Tag d. dt. Einheit	03.10.	Do.	fest	keine
9	Allerheiligen	01.11.	Fr.	fest	keine
10	1. Weihnachtstag	25.12.	Mi.	fest	keine
11	2. Weihnachtstag	26.12.	Do.	fest	keine

Tabelle 2: Wochenfeiertage in Nordrhein-Westfalen im Jahr 2019

Nach Monaten ergeben sich für das Bundesland NW im Jahr 2019 folgende Arbeitstage:

Jan	Feb	Mrz	Apr	Mai	Jun	Jul	Aug	Sep	Okt	Nov	Dez
22	20	21	20	21	18	23	22	21	21	20	20

Tabelle 3: Zahl der Arbeitstage in Nordrhein-Westfalen im Jahr 2019 nach Monaten

In Summe ergeben sich *249* Arbeitstage (vgl. dazu Abbildung 2.)

[1] Ein Feiertag wird als *beweglich* bezeichnet, wenn er nicht in jedem Jahr zum gleichen Datum stattfindet. Bewegliche Feiertage sind meistens kirchliche Feiertage. Sie hängen vom Osterdatum ab und haben einen festen Abstand in Kalendertagen zu diesem (vgl. Tabelle 2, Sp. 6)

6 Deutschen Wochentagnamen zurückgeben

Die benutzerdefinierte PHP-Funktion mit dem Namen '***get_week_day_name***' hat die Aufgabe, den deutschen Wochentagnamen des übergebenen Datums zu bestimmen (s. Listing 7). Diese Funktion wird im aufrufenden PHP-Skript zweimal verwendet (s. Listing 4., Pkt. 13 und Pkt. 14). Der Modulo-Operator (%) führt eine Restwertdivision aus (s. unten, Pkt. 5).

```
function get_week_day_name($dateUS) {
// Aufgabe: Den Namen des Wochentags eines Datums
// mit Julianischer Tagesnummer bestimmen.
   // 1. Datum und Zeit auf Deutsch einstellen.
   setlocale(LC_TIME,"de_DE.utf8");
   date_default_timezone_set('Europe/Berlin');
   // 2. Wochentage in Array speichern.
   $weekDays = array('Sonntag','Montag','Dienstag','Mittwoch','Donnerstag','Freitag','Samstag');
   // 3. Übergebenes Datum aufspalten.
   list($jahr,$monat,$tag) = explode('-',$dateUS);
   // 4. Julianische Tagesnummer ermitteln.
   $jdn = gregoriantojd($monat,$tag,$jahr);
   // 5. Nummer des Wochentags berechnen.
   $DoW = (1 + $jdn) % 7;
   // 6. Rückgabe: Name des Wochentags und Datum mit dt. Datumsformat (tt.mm.jjjj).
   return $weekDays[$DoW].", den ".Date('d.m.Y',strtotime($dateUS));
}
```

Listing 7: Deutschen Wochentagnamen eines Datums zurückgeben

7 Ausblick

Die Beantwortung der Frage „*Auf welches Datum fällt der nächste Arbeitstag*?“ oder „*Auf welches Datum fällt der nächste arbeitsfreie Tag*“ kann mit den präsentierten PHP-Funktionen (s. Tabelle 1) ziemlich leicht bewerkstelligt werden: Statt einer *Zählschleife* (siehe Listing 4, Pkt. 9) bietet sich die Verwendung einer *Do-While-Schleife* an: Sie wird solange ausgeführt, bis die gestellte Bedingung erfüllt ist.[2]

[2] Das bedeutet, dass der Schleifenkörper mindestens einmal durchlaufen wird.

8 Literatur

[1] V. Thormählen, „Fehlerbehandlung bei PHP-Formularen ohne Neueingabe von Daten,“ [Online]. Available: https://docplayer.org/106996948-Fehlerbehandlung-bei-php-formularen-ohne-neueingabe-von-daten-inhalt-i-abbildungen-i-listings-i-1-problemstellung.html. [Zugriff am 01 05 2019].

[2] o. V., „Gesetzliche Feiertage in Deutschland,“ 28 04 2019 . [Online]. Available: https://de.wikipedia.org/wiki/Gesetzliche_Feiertage_in_Deutschland. [Zugriff am 29 05 2019].

[3] o. V., „ PHP Kalender Teil 1, Feiertage berechnen,“ o. J.. [Online]. Available: https://calumoth.de/wordpress/php-kalender-teil-1-feiertage-berechnen-103/. [Zugriff am 29 05 2019].

[4] o. V., „Arbeitstage 2019 Nordrhein-Westfalen,“ [Online]. Available: https://www.schulferien.org/Arbeitstage/Arbeitstage_2019_Nordrhein_Westfalen.html. [Zugriff am 29 05 2019].