

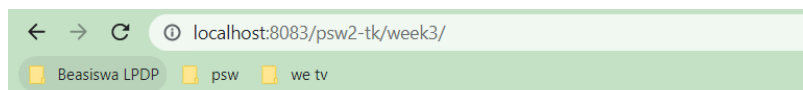
## PHP Tingkat Menengah (Intermediate)

### Membuat template halaman dengan Include dan Require

Ciptakan sebuah file **index.php** dengan konten sebagai berikut:

```
index.php > html
1  <!--
2  Nama   :
3  NIM    :
4  Kelas  :
5  -->
6
7  <!DOCTYPE html>
8  <html lang="en">
9  <head>
10     <meta charset="UTF-8">
11     <meta name="viewport" content="width=device-width, initialscale=1.0">
12     <title>Document</title>
13 </head>
14 <body>
15     <header>
16     | <h1> PABI itu Mudah </h1>
17     </header>
18     <nav>
19     | <li><a href ="index.php"> Home </a></li>
20     | <li><a href ="tentang.php"> Tentang </a></li>
21     | <li><a href ="kontak.php"> Kontak </a></li>
22     </nav>
23     <main>
24     | Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy
25     </main>
26     <footer>
27     | &copy; PABI for Fun 2022
28     </footer>
29 </body>
30 </html>
```

Output:



## PABI itu Mudah

- [Home](#)
- [Tentang](#)
- [Kontak](#)

Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy  
© PABI for Fun 2022

Selanjutnya, Ciptakan sebuah file **tentang.php** dengan konten sebagai berikut:

```

tentang.php > html
1  <!--
2  Nama      :
3  NIM       :
4  Kelas     :
5  -->
6
7  <!DOCTYPE html>
8  <html lang="en">
9  <head>
10     <meta charset="UTF-8">
11     <meta name="viewport" content="width=device-width, initial-scale=1.0">
12     <title>Document</title>
13 </head>
14 <body>
15     <header>
16     | <h1> PABI itu Mudah </h1>
17     </header>
18     <nav>
19     | <li><a href ="index.php"> Home </a></li>
20     | <li><a href ="tentang.php"> Tentang </a></li>
21     | <li><a href ="kontak.php"> Kontak </a></li>
22     </nav>
23     <main>
24     | ini adalah halaman informasi tentang PABI
25     </main>
26     <footer>
27     | &copy; PABI for Fun 2022
28     </footer>
29 </body>
30 </html>

```

Selanjutnya, Ciptakan sebuah file **kontak.php** dengan konten sebagai berikut:

```

kontak.php > html
1  <!--
2  Nama      :
3  NIM       :
4  Kelas     :
5  -->
6
7  <!DOCTYPE html>
8  <html lang="en">
9  <head>
10     <meta charset="UTF-8">
11     <meta name="viewport" content="width=device-width, initial-scale=1.0">
12     <title>Document</title>
13 </head>
14 <body>
15     <header>
16     | <h1> PABI itu Mudah </h1>
17     </header>
18     <nav>
19     | <li><a href ="index.php"> Home </a></li>
20     | <li><a href ="tentang.php"> Tentang </a></li>
21     | <li><a href ="kontak.php"> Kontak </a></li>
22     </nav>

```

```

23     <main>
24     <form action="">
25         <label for="fname">First name:</label><br>
26         <input type="text" id="fname" name="fname" value=""><br>
27         <label for="lname">Last name:</label><br>
28         <input type="text" id="lname" name="lname" value=""><br><br>
29         <input type="submit" value="Submit"> <br><br>
30     </form>
31 </main>
32 <footer>
33     &copy; PABI for Fun 2020
34 </footer>
35 </body>
36 </html>

```

Pada program diatas anda akan dapat mengakses konten dari halaman lain pada halaman yang sama yaitu **index.php**.

Bagaimana seandainya anda ingin mengganti sebuah baris program yang sama disetiap program anda. Contohnya saja baris program **"&copy; PABI for Fun 2022"** ingin diubah menjadi **"&copy; PABI for Fun 2023"** . apa yang harus user lakukan jika hendak merubah bagian ini pada program lain? Solusinya adalah anda harus menggantinya pada setiap halaman (**index.php**, **tentang.php** dan **kontak.php**). Tentunya ini bukan lah program yang efisien dan melanggar asas **DRY (Don't Repeat Your self)**

Maka solusi terbaiknya, PHP menyediakan fungsi **include()** , **require()** dan **require\_once()**. Mari kitapraktikumkan sekarang.

Pindahkan/cut potongan code dari **index.php** berikut kedalam satu file baru dengan nama: **header.php**

```

header.php > html > body > nav
1  <!--
2  Nama      :
3  NIM       :
4  Kelas    :
5  -->
6
7  <!DOCTYPE html>
8  <html lang="en">
9  <head>
10     <meta charset="UTF-8">
11     <meta name="viewport" content="width=device-width, initial-scale=1.0">
12     <title>Document</title>
13 </head>
14 <body>
15     <header>
16         <h1> PABI itu Mudah </h1>
17     </header>
18     <nav>
19         <li><a href ="index.php"> Home </a></li>
20         <li><a href ="tentang.php"> Tentang </a></li>
21         <li><a href ="kontak.php"> Kontak </a></li>
22     </nav>

```

Pindahkan/cut potongan code dari **index.php** berikut kedalam satu file baru dengan nama: **footer.php**

```

❯ footer.php > ...
1  <!--
2  Nama    :
3  NIM     :
4  Kelas   :
5  -->
6
7  <footer>
8      &copy; PABI for Fun 2022
9  </footer>
10 </body>
11 </html>

```

sehingga konten pada file **index.php** menjadi seperti berikut:

```

❯ index.php > ...
1  <!--
2  Nama    :
3  NIM     :
4  Kelas   :
5  -->
6
7  <main>
8      Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy
9  </main>
10

```

Jika anda ingin menampilkan konten **header.php** pada halaman **index.php**, anda cukup menggunakan method **include()** atau **require()** seperti berikut:

```

1  <!--
2  Nama    :
3  NIM     :
4  Kelas   :
5  -->
6
7  <?php
8  include ('header.php');
9  ?>
10
11 <main>
12     Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy
13 </main>
14
15 <?php
16 include ('footer.php');
17 ?>

```

Klik **hyperlink** Home pada website, jika berhasil anda akan melihat konten header.php akan tertampildada web-browser. Ditandai dengan tampilnya header: **“PABI itu Mudah”**.

Selanjutnya silahkan ubah program anda sesuai dengan konten **index.php** dan tambahkan **include('header.php');** kedalam file php anda yang lain (**tentang.php** dan **kontak.php**) dan sehingga perubahannya menjadi seperti berikut pada **tentang.php**:

```
tentang.php > ...
1  <!--
2  Nama      :
3  NIM       :
4  Kelas     :
5  -->
6
7  <?php
8  include ('header.php');
9  ?>
10
11      <main>
12          ini adalah halaman informasi tentang PABI
13      </main>
14
15  <?php
16  include ('footer.php');
17  ?>
```

Ubahlah konten pada **kontak.php** menjadi seperti berikut:

```
kontak.php > ...
1  <!--
2  Nama      :
3  NIM       :
4  Kelas     :
5  -->
6
7  <?php
8  include ('header.php');
9  ?>
10
11      <main>
12          <form action="">
13              <label for="fname">First name:</label><br>
14              <input type="text" id="fname" name="fname" value=""><br>
15              <label for="lname">Last name:</label><br>
16              <input type="text" id="lname" name="lname" value=""><br><br>
17              <input type="submit" value="Submit"> <br><br>
18          </form>
19      </main>
20
21  <?php
22  include ('footer.php');
23  ?>
```

sekarang user akan ditunjukkan pada praktikum ini, perbedaan antara **include()** dan **require()**.

Perbedaannya adalah:

- ❓ **Include():** error akan ditampilkan dari baris program yang salah, tetapi baris program berikutnya masih tetap dieksekusi hingga baris program terakhir.
- ❓ **require():** error akan ditampilkan dari baris program yang salah, tetapi baris program berikutnya tidak akan dieksekusi dan program dipaksa berhenti. Biasanya digunakan untuk form yang bersifat mandatory (wajib) diisi, misalnya halaman **login** dan **register** sebelum user diarahkan ke halaman **utama/dashboard**.

Dikarenakan bagian header dan footer bersifat wajib, maka anda disarankan menggunakan **require()**. Silahkan ubah semua konten pada file **.php** anda dengan **require()**.

Saya akan mencontohkan bagaimana perbedaan antara include() dan require() secara nyata.

Asumsikan anda memanggil konten pada file dengan nama yang salah. Misalnya, seharusnya nama file adalah **header.php** menjadi **headers.php** dan kombinasikan dengan include() dan require().

#### Contoh Include():

```
1  <!--
2  Nama      :
3  NIM       :
4  Kelas     :
5  -->
6
7  <?php
8  include ('headers.php');
9  ?>
10
11  <main>
12      Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy
13  </main>
14
15  <?php
16  include ('footer.php');
17  ?>
```

#### Output:

**Warning:** include(headers.php): Failed to open stream: No such file or directory in C:\xampp\htdocs\psw2-tk\week3\index1.php on line 8

**Warning:** include(): Failed opening 'headers.php' for inclusion (include\_path='C:\xampp\php\PEAR') in C:\xampp\htdocs\psw2-tk\week3\index1.php on line 8

Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy  
© PABI for Fun 2022

Web-browser akan menampilkan error dari baris program include ('headers.php');

Namun baris program selanjutnya masih tetap dieksekusi, yaitu:

```
10
11  <main>
12      Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy
13  </main>
14
15  <?php
16  include ('footer.php');
17  ?>
```

Mari kita bandingkan dengan menggunakan method **require()**, web-browser akan menampilkan error dan tidak akan mengeksekusi baris program selanjutnya dan program dipaksa berhenti.

#### Contoh require():

```

index1.php > ...
1  <!--
2  Nama      :
3  NIM       :
4  Kelas     :
5  -->
6
7  <?php
8  require ('headers.php');
9  ?>
10
11      <main>
12      Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy
13      </main>
14
15  <?php
16  include ('footer.php');
17  ?>

```

## Output

**Warning:** require(headers.php): Failed to open stream: No such file or directory in C:\xampp\htdocs\psw2-tk\week3\index1.php on line 8

**Fatal error:** Uncaught Error: Failed opening required 'headers.php' (include\_path='C:\xampp\php\PEAR') in C:\xampp\htdocs\psw2-tk\week3\index1.php:8 Stack trace: #0 {main} thrown in C:\xampp\htdocs\psw2-tk\week3\index1.php on line 8

Jika menggunakan method **require()**, error ditampilkan oleh browser dan baris program berikutnya tidak akan dieksekusi dan program dipaksa berhenti jika terdapat error.

## Require\_once() dan include\_once()

Require\_once () dan include\_once () digunakan untuk mencegah mengulang template sehingga template yang ditampilkan persis tepat satu sekali.

### Studi kasus: program index.php

```

7  <?php
8  require ('header.php');
9  ?>
10
11  <?php
12  require ('header.php');
13  ?>
14
15      <main>
16      Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy
17      </main>
18
19  <?php
20  require ('footer.php');
21  ?>

```

Asumsikan anda tidak sengaja melakukan `require()` halaman `header.php` pada halaman `index.php` sebanyak 2 kali. Maka akan terdapat bug pada program dengan halaman yang berulang seperti output berikut.



Solusinya anda cukup menggunakan method `require_once()`:

```
week3 > index1.php > ...
1  <!--
2  Nama      :
3  NIM       :
4  Kelas     :
5  -->
6
7  <?php
8  require_once ('header.php');
9  ?>
10
11 <?php
12 require_once ('header.php');
13 ?>
14
15     <main>
16     | Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy
17     </main>
18
19 <?php
20 require_once ('footer.php');
21 ?>
```

Lihat apa yang terjadi?





## PABI itu Mudah

- [Home](#)
- [Tentang](#)
- [Kontak](#)

Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy  
© PABI for Fun 2022

Template halaman **header.php** dan **footer.php** hanya ditampilkan tepat satu kali walaupun **require\_once()** ditulis berulang. Silahkan ubah semua file **PHP** anda dengan menggunakan **method require\_once()**.

### Explode dan Implode

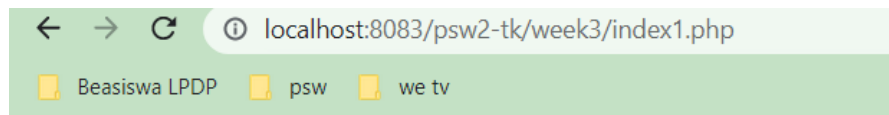
Perbedaan Explode dan Implode adalah:

- ❓ Explode: Mengubah format data dari **string** menjadi bentuk **array**.
- ❓ Implode: Kebalikannya, Mengubah format data dari **array** kedalam bentuk **string**. Contoh

Program **implode** pada **index.php**:

```
week3 > index1.php > ...
1  <!--
2  Nama      :
3  NIM       :
4  Kelas     :
5  -->
6
7  <?php
8  require_once ('header.php');
9  ?>
10
11  <main>
12  | Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy
13  </main>
14
15  <?php
16  $pekerjaan = ["guru", "penyihir", "pencuri", "pendeta", "ustadz"];
17  echo implode(" | ", $pekerjaan) . '<br>';
18  var_dump(implode(" | ", $pekerjaan));
19  echo '<br>';
20  ?>
21
22  <?php
23  require_once ('footer.php');
24  ?>
```

**Output:**



# PABI itu Mudah

- [Home](#)
- [Tentang](#)
- [Kontak](#)

Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy  
guru | penyihir | pencuri | pendeta | ustadz

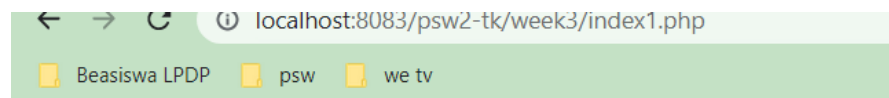
string(44) "guru | penyihir | pencuri | pendeta | ustadz"

© PABI for Fun 2022

Contoh Program **Explode** pada **index.php**:

```
week3 > index1.php > ...
1  <!--
2  Nama      :
3  NIM       :
4  Kelas     :
5  -->
6
7  <?php
8  require_once ('header.php');
9  ?>
10
11  <main>
12  | Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy
13  </main>
14
15  <?php
16  $pelajaran = "PABI SISOP SISTAN PLSJ ANJAR";
17  print_r(explode(" ", $pelajaran, 3));
18  ?>
19
20  <?php
21  require_once ('footer.php');
22  ?>
```

Output:



# PABI itu Mudah

- [Home](#)
- [Tentang](#)
- [Kontak](#)

Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy  
Array ( [0] => PABI [1] => SISOP [2] => SISTAN PLSJ ANJAR )

© PABI for Fun 2022

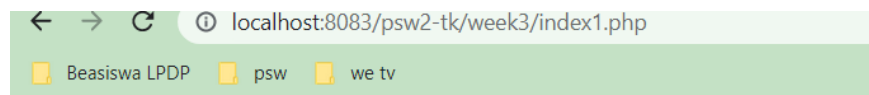
## Fungsi date

Referensi: <https://www.php.net/manual/en/function.date.php>

Contoh Program menggunakan fungsi date() pada index.php:

```
7 <?php
8 require_once ('header.php');
9 ?>
10
11 <main>
12 | Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy
13 </main>
14
15 <?php
16 $pekerjaan = ["guru", "penyihir", "pencuri", "pendeta", "ustadz"];
17 echo implode(" | ", $pekerjaan) . '<br>';
18 var_dump(implode(" | ", $pekerjaan));
19 echo '<br>';
20
21 $pelajaran = "PABI SISOP SISTAN PLSJ ANJAR";
22 print_r(explode(" ", $pelajaran, 3));
23 ?>
24
25 <h2>Fungsi Date</h2>
26
27 <?php
28 echo date('d - M - Y');
29 ?>
30
31 <?php
32 require_once ('footer.php');
33 ?>
```

Output:



## PABI itu Mudah

- [Home](#)
- [Tentang](#)
- [Kontak](#)

Selamat datang di mata kuliah PABI, tempat belajar PABI yang happy

guru | penyihir | pencuri | pendeta | ustadz

string(44) "guru | penyihir | pencuri | pendeta | ustadz"

Array ( [0] => PABI [1] => SISOP [2] => SISTAN PLSJ ANJAR )

## Fungsi Date

06 - Feb - 2023

© PABI for Fun 2022

Sekarang anda dapat membuat konten **tahun** pada halaman **footer.php** anda menjadi dinamis menggunakan fungsi **date**.

Silahkan ubah “2022” yang sebelumnya diketik manual, menjadi otomatis menggunakan fungsi **date()** seperti berikut:

```
6
7     <footer>
8         &copy; PABI for Fun
9
10        <?php
11            echo date ('M-Y')
12        ?>
13
14    </footer>
15 </body>
16 </html>|
```

Ouput:



### Trim() dan strip\_tags

Fungsi trim() yang digunakan untuk menghapus spasi (whitespace) di awal dan akhir sebuah stringPHP. Fungsi trim() juga memiliki 2 varian lain yakni ltrim() dan rtrim().

Pengertian Fungsi trim()

Secara default bawaan PHP, fungsi trim() digunakan untuk menghapus spasi atau karakter whitespacedari sebuah string. Karakter spasi yang akan dihapus bisa berada di awal maupun di akhir string.

Dalam prakteknya, fungsi trim() sering digunakan untuk ‘membersihkan’ hasil input form dari karakter spasi yang sengaja atau tidak sengaja ditambahkan pengguna.

Berikut adalah contoh penggunaan dasar fungsi trim() di dalam PHP:

```

7  <?php
8  $nama = " andi ";
9  $trim_nama = trim($nama);
10 echo $trim_nama; // "andi"
11 ?>

```

Karena di HTML whitespace atau spasi tidak akan ditampilkan, penerapan kode diatas tidak terlalu jelas efeknya. Fungsi trim() akan lebih terlihat jika digunakan dalam operasi perbandingan, seperticontoh berikut:

```

6
7  <?php
8  $nama = "andi ";
9  $nama_juga = " andi";
10 if ($nama == $nama_juga) {
11     echo "Nama Sama";
12 }
13 else {
14     echo "Nama Beda";
15 }
16 // hasil: Nama Beda
17 ?>
18

```

Dalam operasi perbandingan diatas, tambahan sebuah spasi diakhir variabel \$nama, yakni “andi ” akan membuat operasi perbandingan menghasilkan nilai FALSE, sehingga hasil akhirnya adalah “Tidak Sama”.

Dengan menambahkan fungsi trim(), kode program diatas akan menghasilkan nilai TRUE, karena spasi yang ada baik diawal dan diakhir string akan dihapus terlebih dahulu:

```

7  <?php
8  $nama = "andi ";
9  $nama_juga = " andi";
10 if (trim($nama) == trim($nama_juga)) {
11     echo "Nama Sama";
12 }
13 else {
14     echo "Nama Beda";
15 }
16 // hasil: Nama Sama
17 ?>
18

```

Selain menghapus karakter spasi, fungsi trim() juga akan menghapus 5 karakter whitespace lainnya, seperti tab, new line, carriage return (karakter enter), null-byte, dan vertical tab. Dalam kode karakterASCII, ke-6 karakter ini adalah sebagai berikut:

- ❓ " " (ASCII 32 (0x20)), : karakter spasi.
- ❓ "\t" (ASCII 9 (0x09)), : karakter tab.

- ❓ “\n” (ASCII 10 (0x0A)), : karakter new line (line feed).
- ❓ “\r” (ASCII 13 (0x0D)), : karakter carriage return.
- ❓ “\0” (ASCII 0 (0x00)), : karakter NULL-byte.
- ❓ “\x0B” (ASCII 11 (0x0B)), : karakter vertical tab. Berikut contoh penggunaannya:

```

6
7 <?php
8     $nama = "\t \t andi \n \r";
9     $nama_juga = "    andi \t";
10    if (trim($nama) == trim($nama_juga)) {
11        echo "Nama Sama";
12    }
13    else {
14        echo "Nama Beda";
15    }
16    // hasil: Nama Sama
17    ?>
18

```

Untuk menulis karakter “tab”, tidak tersedia tombol khusus di dalam keyboard, oleh karena itu kita menggunakan escape karakter untuk tab, yakni “\t”.

Fungsi trim() juga memiliki argumen kedua yang bersifat opsional. Argumen kedua ini bertipe string yang jika ditulis akan ditambahkan kedalam daftar karakter yang ikut dihapus, atau istilah teknisnya disebut dengan **character\_mask**. Langsung saja kita lihat contoh penggunaannya:

```

6
7 <?php
8     $nama = "__andi__";
9     $trim_nama = trim($nama);
10    echo $trim_nama; // "__andi__"
11    echo "<br>";
12
13    $trim_nama = trim($nama, "_");
14    echo $trim_nama; // "andi"
15    ?>
16

```

Lebih jauh lagi, character\_mask ini mendukung penulisan range atau jangkauan karakter, yang ditulis dengan “awal..akhir”.

Sebagai contoh, jika saya memiliki beberapa string yang dimulai dengan angka seperti: “1 kelereng”, “2 buah”, “3 orang”, saya bisa menggunakan fungsi **trim()** untuk menghapus seluruh angka awalan ini. Berikut contohnya:

```

6
7 <?php
8     $kata = "1 kelereng";
9     $trim_kata = trim($kata,"0..9");
10    echo $trim_kata; // "kelereng"
11    echo "<br>";
12
13    $kata = "2 buah";
14    $trim_kata = trim($kata,"0..9");
15    echo $trim_kata; // "buah"
16    echo "<br>";
17
18    $kata = "3 orang";
19    $trim_kata = trim($kata,"0..9");
20    echo $trim_kata; // "orang"
21    ?>

```

Fungsi **trim(\$kata,"0..9")** berarti: hapus whitespace yang ada di awal dan akhir string \$kata, dan hapus karakter 0, 1, 2, 3 s/d 9 di awal dan diakhir string. Fitur untuk menghapus karakter tertentu ini cukup berguna ketika kita ingin 'membersihkan' sebuah string dari karakter yang tidak diinginkan.

### Pengertian Fungsi rtrim() dan ltrim()

Fungsi **rtrim()** dan **ltrim()** adalah bentuk lain dari fungsi **trim()**, tapi hanya akan menghapus karakter whitespace yang ada disisi kanan (untuk **rtrim()**) dan di sisi kiri string (untuk **ltrim()**). Kedua fungsi ini juga bisa ditambahkan argumen ketiga seperti halnya fungsi **trim()**.

Berikut contoh penggunaan fungsi **rtrim()** dan **ltrim()** dalam PHP:

```

6
7 <?php
8     $nama = " andi ";
9
10    $rtrim_nama = rtrim($nama);
11    echo $rtrim_nama; // " andi"
12    echo "<br>";
13
14    $ltrim_nama = ltrim($nama);
15    echo $ltrim_nama; // "andi "
16    echo "<br>";
17
18    $nama1 = "__andi__";
19
20    $rtrim_nama1 = rtrim($nama1,"_");
21    echo $rtrim_nama1; // "__andi"
22    echo "<br>";
23
24    $ltrim_nama1 = ltrim($nama1,"_");
25    echo $ltrim_nama1; // "andi__"
26    ?>
27

```

Dapat terlihat bahwa fungsi **rtrim()** dan fungsi **ltrim()** hanya akan menghapus karakter pada satu sisi saja.

Fungsi **trim()**, **rtrim()** dan **ltrim()** yang kita pelajari disini cocok digunakan untuk operasi 'pembersihan' string, ini umumnya digunakan untuk menfilter hasil inputan form dari pengguna (user).

### Fungsi **strip\_tags()** dengan **htmlentities()**

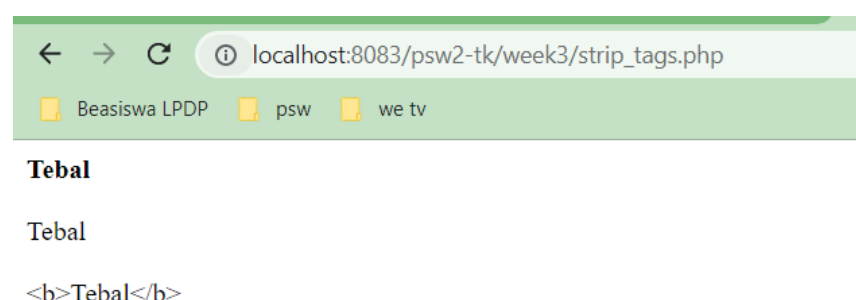
Jika anda melihat sebuah website pada bagian kolom komentar, sering para pengunjung meletakkan nama urlnya dengan menggunakan tag HTML **<a href="namadomain">Kalimat</a>**, dengan maksud agar visitor lain dapat mengklik linknya dan menuju suatu situs tertentu. Nah sekarang bagaimana agar setiap pengunjung yang mengisi komentar hanya dapat mengisi teks biasa, jika memasukkan tag HTML maka akan dihapus bagian tag HTMLnya aja.

Supaya isi dari kolom komentar dianggap teks biasa, ada dua teknik yang dapat kita lakukan. Yang pertama dengan menggunakan fungsi **strip\_tags()**, yang kedua dengan menggunakan fungsi **htmlentities()**.

Perbedaan antara fungsi **strip\_tags()** dengan **htmlentities()** ditunjukkan oleh contoh dibawah ini :

```
week3 > strip_tags.php > ...
1  <?php
2      $kalimat = "<b>Tebal</b>";
3      echo $kalimat."<br><br>";
4      pakai_strip_tags();
5      pakai_htmlentites();
6
7      function pakai_strip_tags()
8      {
9          GLOBAL $kalimat;
10         $filter1 = strip_tags($kalimat);
11         echo $filter1."<br><br>";
12     }
13     function pakai_htmlentites()
14     {
15         GLOBAL $kalimat;
16         $filter2 = htmlentities($kalimat);
17         echo $filter2;
18     }
19  ?>
```

Outputnya :



localhost:8083/psw2-tk/week3/strip\_tags.php

Beasiswa LPDP   psw   we tv

**Tebal**

Tebal

<b>Tebal</b>



Seperti yang kita lihat, apabila terdapat sebuah kalimat yang menggunakan tag HTML `<b></b>`, maka kalimat tersebut akan ditampilkan dengan huruf tebal, yaitu : Tebal

Nah karena kita gunakan fungsi `strip_tags()`, maka tag HTML tersebut akan dihapus, sehingga hanya ditampilkan kalimat tanpa mengalami penebalan, yaitu : Tebal.

Tetapi jika kita menggunakan fungsi `htmlentities()`, maka tag HTML `<b><b>` akan dianggap teks biasa, sehingga akan ditampilkan : `<b>Tebal</b>`.

Dalam pembahasan berikutnya kita akan fokus tata cara penggunaan fungsi `strip_tags()` berserta contoh penerapannya.

### Pengenalan

Seperti yang dijelaskan diatas, fungsi **`strip_tags()`** bertujuan untuk menghilangkan tag HTML. Fungsi ini mulai diperkenalkan pada **PHP versi 4+** dan masih digunakan hingga sekarang. Syntax dari penulisan fungsi `substr()` :

`strip_tags(string, allowable_tags)`

Keterangan:

- ❓ `string` mengacu kepada sebuah kalimat yang akan diperiksa (Required)
- ❓ `allowable_tags` mengacu kepada tag HTML yang diizinkan (Optional)

### Menghilangkan semua tag HTML

Misalkan kita memiliki sebuah kalimat : Pusat Ilmu Secara Detil

Jika kita melihat kalimat tersebut, maka yang dibold adalah Pusat dan Detil, sedangkan yang italic adalah Ilmu Secara.

Untuk membuat bold dalam HTML kita menggunakan tag `<b></b>` Untuk membuat italic dalam HTML kita menggunakan tag `<i></i>`

Nah sekarang kita akan tunjukkan bagaimana menghilangkan kedua tag HTML tersebut, yaitu: tag `<b></b>` serta tag `<i></i>`, sehingga kalimat yang ditampilkan adalah:

Pusat Ilmu Secara Detil tanpa ada penebalan (bold) pada Pusat dan Detil serta tanpa ada pemiringan pada kalimat Ilmu Secara.

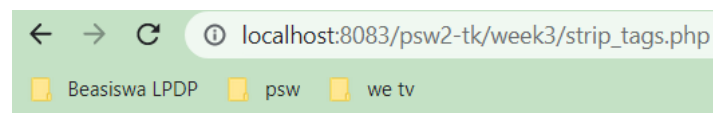
Perhatikan Contoh penggunaannya untuk memahami lebih lanjut:

```

week3 > strip_tags.php > ...
1  <?php
2      tanpa_strip_tags();
3      memakai_strip_tags();
4      function tanpa_strip_tags()
5      {
6          $kalimat = "<b>Pusat</b> <i>Ilmu Secara</i> <b>Detil</b>";
7          echo $kalimat;
8          echo"<br><br>";
9      }
10     function memakai_strip_tags()
11     {
12         $kalimat = "<b>Pusat</b> <i>Ilmu Secara</i> <b>Detil</b>";
13         echo strip_tags($kalimat);
14     }
15     ?>

```

Outputnya akan ditunjukkan oleh gambar dibawah ini:



**Pusat Ilmu Secara Detil**

Pusat Ilmu Secara Detil

### Menghilangkan tag HTML tertentu

Kita masih menggunakan kalimat diatas yaitu: Pusat Ilmu Secara Detil

Sekarang kita akan menghilangkan bagian yang ditebalkan pada kata Pusat dan Detil. Untuk itu kita akan mengizinkan tag `<i></i>` dan membuang tag `<b></b>`.

Sehingga output yang kita harapkan adalah :

Pusat Ilmu Secara Detil

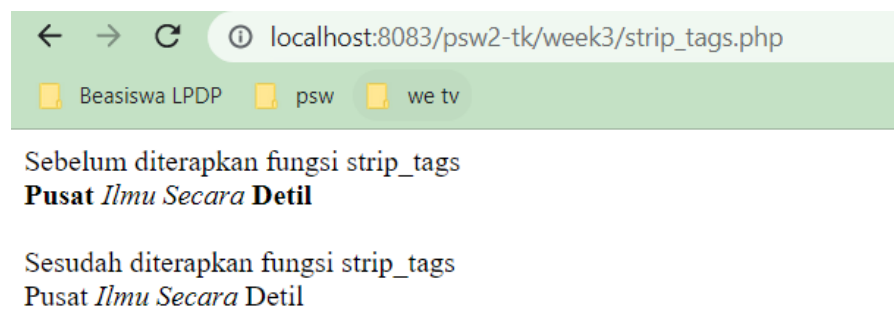
Perhatikan contoh berikut untuk memahami bagaimana menghilangkan tag HTML tertentu:

```

week3 > strip_tags.php > ...
1  <?php
2      tanpa_strip_tags();
3      memakai_strip_tags();
4      function tanpa_strip_tags()
5      {
6          echo "Sebelum diterapkan fungsi strip_tags". "<br>";
7          $kalimat = "<b>Pusat</b> <i>Ilmu Secara</i> <b>Detil</b>";
8          echo $kalimat;
9          echo "<br><br>";
10     }
11     function memakai_strip_tags()
12     {
13         echo "Sesudah diterapkan fungsi strip_tags". "<br>";
14         $kalimat = "<b>Pusat</b> <i>Ilmu Secara</i> <b>Detil</b>";
15         echo strip_tags($kalimat, "<i>");
16     }
17     ?>

```

Outputnya akan ditunjukkan oleh Gambar dibawah ini:



*"At forty, I was too old to work as a programmer myself anymore; writing code is a young person's job."*

**Michael Crichton**