**Fungsi , Modularisasi dan Form dalam PHP**

**Pengertian Fungsi dalam PHP**

Fungsi adalah kode program yang dibuat khusus untuk mempersingkat sebuah aksi yang ingin dilakukan dalam sebuah penulisan kode program. Dan fungsi juga dapat di panggil berkali – kali untuk melakukan tugas tertentu dalam penulisan kode program. Fungsi juga merupakan salah satu Teknik pemrograman modular. Sebuah aplikasi besar yang disusun dari modul – modul yang berupa sebuah fungsi atau prosedur. Fungsi berisi sekelompok kode dengan tugas dan tujuan spesifik. Fungsi tidak akan dieksekusi ketika program dijalankan. Fungsi hanya akan dieksekusi jika dilakukan pemanggilan terhadapnya. Pemanggilan dapat dilakukan dari mana saja dalam program. Fungsi juga dapat dibuat sendiri untuk melakukan sebuah aksi. Keuntungan Teknik ini, modul – modul yang dibuat dapat digunakan berkali – kali (*reuse*)

Terdapat dua hal pokok yang dapat dilakukan pada sebuah function, yaitu :

1. Mendefinisikan fungsi sendiri, jika fungsi yang diinginkan belum disediakan oleh PHP.
2. Memanggil fungsi, baik built – in function dan user – defined function cara memanggilnya sama yaitu menuliskan nama fungsi kemudian diikuti tanda kurung, misalkan memanggil fungsi cetak, maka penulisannya :

**Cetak();**

**Membuat fungsi**

Sebuah fungsi dibuat dengan aturan sintaks, yaitu:

*Function namaFungsi() {*

*Kode – kode yang akan dieksekusi;*

*}*

Beberapa petunjuk dalam membuat sebuah fungsi :

* Nama fungsi yang menggambarkan fungsinya
* Nama fungsi dimulai dengan huruf atau garis bawah (*underscore*), tidak boleh menggunakan angka.

**Jenis – jenis Fungsi**

1. **Fungsi tanpa parameter**

Fungsi tanpa parameter adalah fungsi dimana pembuatan dan pengambilan fungsi dilakukan tanpa menggunakan parameter. Berikut struktur fungsi tanpa parameter, yaitu:

*Function namaFungsi()*

 *{*

*Function body;*

*}*

Berikut adalah penjelasan dari gambar di atas:

1. Function

Function merupakan kata yang harus di tulis ketika akan membuat suatu fungsi

1. Nama fungsi

Nama fungsi adalah nama yang akan diberikan oleh si pemrograman dengan catatan nama fungsi tidak boleh diawali angka dan tidak boleh sama dengan nama function

1. Function body

Function body berisi body program yang akan di proses didalam fungsi

Sintaks :

1. Sintaks pemanggilan fungsi

*namaFungsi() ;*

1. Sintaks Lengkap pemanggilan fungsi

*Function namaFungsi()*

 *{*

*Function body;*

*}*

*namaFungsi()*

Contoh :



Keterangan :

* Pada baris ke 1 dan 10 merupakan tag pembuka dan penutup PHP
* Pada baris ke 2 sampai dengan 8 membuat fungsi dan nama perkalian
* Pada baris ke 9 merupakan perintah PHP pemanggilan fungsi.

Hasil, ketika dijalankan dengan web browser:



1. **Fungsi dengan parameter**

Fungsi dengan parameter adalah fungsi yang memiliki parameter dimana nilai yang dimasukkan kedalam fungsi ini dapat di proses sehingga menghasilkan output. Berikut adalah sintaks fungsi tanpa parameter:

*Function namaFungsi($parameter)*

 *{*

*Function body;*

*}*

Berikut adalah penjelasan dari sintaks fungsi tanpa parameter:

1. Function

Function merupakan kata yang harus di tulis ketika akan membuat suatu fungsi

1. Nama fungsi

Nama fungsi adalah nama yang diberikan oleh pemrogram dengan catatan nama fungsi tidak boleh diawali angka dan tidak boleh sama dengan nama function

1. $parameter

Merupakan nama parameter yang digunakan

1. Function body

Berisi body program yang akan diproses di dalam fungsi

Berikut sintaks cara pemanggilan parameter pada fungsi :

*Namaparameter($parameter1, $parameter2);*

Contoh fungsi tanpa parameter:



Keterangan :

* Pada baris ke 1 dan 8 merupakan tag pembuka dan penutup PHP
* Pada baris ke 2 sampai dengan 6 merupakan tag PHP untuk membuat fungsi perkalian
* Pada baris ke 7 merupakan tag PHP untuk memanggil fungsi perkalian

Hasil yang dijalankan pada halaman web browser :



**Pemanggilan fungsi**

Ketika fungsi sudah dibuat, dapat dijalan dengan beberapa cara panggilan. Pemanggilan suatu fungsi mengikuti pola: nama fungsi lalu diikuti dengan tanda kurung dan nilai parameter yang ada.

Contoh :

*Tambah (10, 20);*

Pada contoh diatas, pemanggilan sebuah fungsi bernama tambah dengan nilai parameter 10 dan 20. Dan dapat juga dibuat dengan tidak menggunakan nilai parameter, maka pemanggilan fungsi, akan seperti:

*Cetak ();*

**Macam – macam fungsi**

Fungsi terbagi menjadi 3 yaitu :

1. **Fungsi built in**

Fungsi ini sudah disediakan oleh PHP dan pemrograman dapat langsung memakainya.

Seperti :

1. Fungsi matematika

Fungsi matermatika seperti : sin, cos, tan, asin, deg2rad, dll

1. Fungsi tanggal & waktu

Fungsi tanggal & waktu, seperti : date, checkdate

1. Fungsi string

Fungsi string, seperti : strlen, strpos, strytolower, strtoupper, substr, dll

 Contoh :

 Fungsi built-in string :

 

 Hasil pada layar browser:

 

1. **Fungsi UDF (*user defined function*)**

Fungsi UDF merupakan fungsi yang dibuat sendiri oleh pemrograma. Dapat diletakkan dimana saja dalam script PHP, tetapi sebaiknya diletakkan paling atas agar memudahkan dalam proses debugging dalam tahap pembuatan dan pengembangan program.

Struktur UDF:

*Function namafungsi ([parameter]) {*

 *Statement1;*

 *Statement2;*

 *……;*

*}*

Pemanggilan UDF :

*$varhasil = &namafungsi([parameter]);*

Penggunaan tanda & tidak mutlak, karena PHP secara otomatis akan menganggap sebagai fungsi karena tidak didahului dengan tanda $.

Contoh fungsi UDF:



Hasil dalam layar browser :



1. **Fungsi external**

**Nilai balik fungsi**

Nilai balik fungsi adalah fungsi yang dapat diatur agar mengembalikan hasil berupa nilai dengan cara menggunakan kata kunci return. Seperti contoh dibawah ini :

*<html>*

*<body>*

*<?php*

 *Function tambah(){*

 *$total = $x + $y;*

 *Return $total;*

*}*

*Echo “$x + $y = “.tambah(5,20);*

*?>*

*</body>*

*</html>*

Nilai yang dikembalikan pada fungsi di atas adalah jumlah dari nilai variabel $x dan $y yang ada di dalam variabel $total.

Hasil dari layer web browser adalah 5 + 20 = 25.

**Variabel lifetime**

**Scope**

Scope, lifetime, visibility adalah berbagai istilah yang sama, menyatakan dimana variabel dapat digunakan dalam program. PHP mengenal dua macam scope yaitu local dan global. Variabel local hanya dapat digunakan dalam blok tempatnya dideklarasi. Variabel global dapat digunakan dimana saja dalam program setelah dideklarasi dan diinisialisasi.

**Local**

Penggunaan scope local, dapat dilihat dari contoh dibawah ini :



Variabel $a hanya berlaku di dalam function testVar, sehingga tidak dapat diakses di luar function.

**Global**

Variabel global dideklarasi di luar function dan dapat digunakan dimana saja dalam program. Variabel global tidak berlaku di dalam function kecuali di panggil dalam keyword global.

Contoh :



**Passing variabel by value**

Passing variabel by value adalah variabel yang dimasukkan ke dalam function dengan cara pass by value. Pass by value membuat Salinan dari variabel yang asli. Sehingga variabel asli tidak terpengaruh.

Contoh :



Fungsi diatas dapat digunakan untuk , seperti dibawah ini :



Hasil dari script diatas, adalah :



Karena variabel angka yang ada dalam fungsi berakhir umumnya pada saat fungsi selesai dijalankan. Dan variabel a yang asli tetap bernilai 7. Untuk menghasilkan efek yang diinginkan, maka dapat menggunakan **pass by reference.**

**pass by reference**

pass by reference, dapat memasukkan variabel yang asli ke dalam fungsi. Cara ini dilakukan, sebagai berikut :



Script diatas dapat digunakan untuk contoh dibawah ini :



Hasil pada layer browser :



**Modularisasi**

 Modularisasi dalam program umum dilakukan dengan sangat diperlukan untuk mempermudah debugging dan pengembangan program. Modularisasi berarti melakukan pembuatan program berdasarkan modul – modul. Setiap modul dikembangkan untuk tujuan atau fungsi khusus, modul dibuat secara umum (general). Modu dapat juga berupa fungsi atau prosedur.

**Require**

Statement require digunakan untuk membaca nilai variabels dalam fungsi – fungsi dari sebuah file lain.

Sintaks statement require adalah :

*Require (namafile);*

 Teknik ini cocok untukmembuat template (pola) yang memudahkan proses pengembangan aplikasi dengan menggunakan pola tampilan. Misal dalam membuat design web yang dilakukan dengan kerja tim, pola dan gambar – gambar yang dibutuhkan dapat dibuat dengan Teknik ini.

 Statement require ini tidak dapat dimasukkan di dalam suatu struktur looping misalnya while atau for. Karena hanya memperbolehkan pemanggilan file yang sama tersebut hanya sekali saja.

**Include**

Statement include akan menyertakan isi suatu file tertentu. Include dapat diletakkan di dalam suatu looping misalkan dalam statement for atau while.

Sintaks penulisan include:

*include (namafile);*

Cara untuk menginclude file PHP, terdapat beberapa cara diantaranya :



**Perbedaan Include dan Require**

|  |  |
| --- | --- |
| **Include** | **Require** |
| Memiliki fungsi yang sama  | Memiliki fungsi yang sama  |
| Hanya akan menghasilkan notice jika file tidak ada | Menghasilkan fatal error jika file tidak ada |
| **\_once** Untuk tidak menampilkan kembali file yang dipanggil jika file tersebut sudah pernah dipanggil sebelumnya  | **\_once** Untuk tidak menampilkan kembali file yang dipanggil jika file tersebut sudah pernah dipanggil sebelumnya |

Contoh modularisasi menggunakan include :

1. Nama file include.php



1. File memanggil file php diatas dan menyimpan dengan nama memanggil.php



1. Hasil :



**FORM dalam PHP**

 Form adalah antarmuka (*user interface)* agar user dapat berkomunikasi dengan sistem aplikasi. Setiap permintaan dari user disebut request dan setiap hasil yang dari pemrosesan oleh server disebut respons.

 Form dalam aplikasi web menggunakan tag <form> dan diakhiri </form>. Dimana tag tersebut mempunyai beberapa atribut antara lain adalah action dan method. Action digunakan untuk mendefinisikan tujuan pengiriman data dari form, dan method digunakan untuk mendefinsikan metode pengiriman apa yang akan digunakan. Method dalam atribut ada 2 jenis yaitu :

1. **POST**

POST adalah proses mengirim data (submit) melalui form yang berasal dari XHTML. Karena PHP adalah Bahasa program atau scripting, maka PHP bisa digunakan untuk menerima data hasil submit tersebut. Data yang diterima nantinya bisa diproses dalam script PHP. Untuk menerima data dari proses submit form dalam PHP, dapat digunakan statement assignment, sebagai berikut ini:

*$namaVariabel = $\_POST[‘nama komponen dalam form’];*

Dalam hal ini, nama variabel boleh sama atau tidak sama dengan nama komponen dalam form. Tetapi, sangat disarankan untuk membuat nama variabel yang berbeda dari nama komponen formnya untuk mencegah hal – hal yang tidak diinginkan, misalkan hacking.

 Contoh :

 Membuat form HTM yang didalamnya terdapat dua komponen input dua buah bilang

 **Input.html**

*<html>*

*<head>*

 *<title> contoh request POST </title>*

*</head>*

*<body>*

 *<h1> input dua bilangan </h1>*

 *<form method = “POST” action = “proses.php”>*

 *Masukkan bilangan 1 : <input type = “text” name = “bil1”/><br/>*

 *Masukkan bilangan 2 : <input type = “text” name=”bil2” /><br/>*

 *<input type = “submit” name =”submit” value = “submit” />*

 *</form>*

*</body>*

*</html>*

pada form di atas dapat dilihat bahwa terdapat dua komponen dalam form yang masing – masing memiliki nama “bil1” dan “bil2” (perhatikan atribut name = “ … “ pada komponen). Keduanya merupakan komponen form berbentuk text box . kemudian perhatikan bagian atribut action = “proses.php”. atribut yang memiliki tombol submit tersebut untuk di klik. Maka data yang diisikan pada form nantinya akan dikirim dan diolah pada script bernama proses.php.

pada script proses.php, dalam script ini, dapat dibaca nilai yang diisikan ke dalam form di atas, lalu menampilkannya ke browser proses.php.

*<?php*

*$bilangan1 = $\_POST[‘bil1’];*

*$bilangan2 = $\_POST[‘bil2’];*

*Echo “<p> anda telah memasukkan bilangan pertama = “.$bilangan1.”</p>”;*

*Echo “<p> anda telah memasukkan bilangan kedua = “.$bilangan2.”</p>”;*

*?>*

Anda juga menggabungkan script di atas dengan struktur XHTML yang lebih lengkap seperti di bawah ini.

Proses.php

*<?php*

*$bilangan1 = $\_POST[‘bil1’];*

*$bilangan2 = $\_POST[‘bil2’];*

*?>*

*<html>*

*<head>*

 *<title> contoh request POST </title>*

*</head>*

*<body>*

 *<h1> input dua bilangan </h1>*

*<?php*

*Echo “<p> anda telah memasukkan bilangan pertama = “.$bilangan1.”</p>”;*

*Echo “<p> anda telah memasukkan bilangan kedua = “.$bilangan2.”</p>”;*

*?>*

*</body>*

*</html>*

Selain cara di atas yaitu menyimpan nilai dari $\_POST[] ke variabel tertentu, dapat langsung menggunakan $\_POST[] untuk diproses tanpa disimpan ke variabel tertentu.

contoh :

*<?php*

*$bilangan1 = $\_POST[‘bil1’];*

*$bilangan2 = $\_POST[‘bil2’];*

*?>*

*<html>*

*<head>*

 *<title> contoh request POST </title>*

*</head>*

*<body>*

 *<h1> input dua bilangan </h1>*

*<?php*

*Echo “<p> anda telah memasukkan bilangan pertama = “.$\_POST[‘bil1’].”</p>”;*

*Echo “<p> anda telah memasukkan bilangan kedua = “.$\_POST[‘bil2’].”</p>”;*

*?>*

*</body>*

*</html>*

Pada contoh pertama, dengan membuat script proses.php hanya sekedar menampilkan dua bilangan yang diinputkan pada form. Selanjutnya pada script berikutnya, misalkan masih tetap sama nama filennya yaitu proses.php akan tetapi digunakan untuk menampilkan hasil penjumlahan dua bilangan yang diinputkan tadi. Adapun konsepnya hampir sama, yaitu pertama baca terlebih dahulu bilangan – bilangannya lalu jumlahkan keduanya dan hasil penjumlahannya ditampilkan.

Contoh :

Proses.php

*<?php*

*$bilangan1 = $\_POST[‘bil1’];*

*$bilangan2 = $\_POST[‘bil2’];*

*$jumlah = $bilangan1 + $bilangan2;*

*?>*

*<html>*

*<head>*

 *<title> contoh request POST </title>*

*</head>*

*<body>*

 *<h1> input dua bilangan </h1>*

*<?php*

*Echo “<p> anda telah memasukkan bilangan pertama = “.$bilangan1.”</p>”;*

*Echo “<p> anda telah memasukkan bilangan kedua = “.$bilangan2.”</p>”;*

*Echo “<p> hasil penjumlahannya adalah “.$jumlah.”</p>”;*

*?>*

*</body>*

*</html>*

Contoh :



Membaca variabel dari form dengan metode POST: $\_POST[“nama\_variabel”]

1. **GET**

Apabila POST request adalah proses merequest data melalui form, maka GET request adalah melalui URL. Terkadang ketika browsing di internet, sering ditemukan URL atau link yang berbentuk misalnya seperti berikut :

 <http://situs.com/script.php?a=4&b=6>

perhatikan bagian URL diatas tersebut, bagian tersebut dinamakan dengan parameter. Konsep GET pada prinsipnya mirip dengan POST. Pada link diatas adalah mensubmit atau mengirim data ke script bernama script.php dan nilai yang dikirim adalah nilai a = 4 dan nilai b = 6. Nilai yang dikirim tersebut nantinya akan dijadikan sebagai input pada proses yang terjadi di script.php.

contoh :

*<?php*

*$bilangan1 = $\_GET[‘bil1’];*

*$bilangan2 = $\_GET[‘bil2’];*

*$jumlah = $bilangan1 + $bilangan2;*

*?>*

*<html>*

*<head>*

 *<title> contoh request GET </title>*

*</head>*

*<body>*

 *<h1> input dua bilangan </h1>*

*<?php*

*Echo “<p> anda telah memasukkan bilangan pertama = “.$bilangan1.”</p>”;*

*Echo “<p> anda telah memasukkan bilangan kedua = “.$bilangan2.”</p>”;*

*Echo “<p> hasil penjumlahannya adalah “.$jumlah.”</p>”;*

*?>*

*</body>*

*</html>*

Contoh :

****

Membaca variabel dari form dengan metode GET: $\_GET[“nama\_variabel”]

Ada beberapa komponen field yang bisa digunakan di dalam form, antara lain adalah :

1. <input>, untuk jenis inputan dalam bentuk isian satu baris.
2. <textarea>, untuk jenis inputan dalam bentuk multi baris
3. <option>, untuj jenis inputan pilihan

**Validasi FORM**

Input user sebaiknya divalidasi pada sisi browser menggunakan script klien, misalnya javascript, bila memungkinkan. Ini akan mempercepat proses validasi dan mengurangi komunikasi dengan server.

Jika input user akan disimpan ke dalam database, perlu dipertimbangkan melakukan proses validasi di server. Cara yang disarankan dalam melakukan proses validasi di server adalah mengirimkan data form ke form itu sendiri. Maka isi dari atribut action form berupa nama file dari form itu sendiri.

Pada contoh, isi dari atribut action yaitu berisi welcome.php diganti form.php. kemudian isi file welcome.php ditulis dibawah kode yang ada pada file form.php sehingga isi dari file form.php akan menjadi :

*<html>*

*<body>*

*<form action = “welcome.php” method = “POST”>*

 *Name : <input type = “text” name=”nama” / >*

 *Age : <input type = “text” name= “usia” />*

 *<input type = “submit”>*

*</form>*

 *Welcome*

 *<?php*

 *echo $\_POST[“nama”];*

 *?>! <br />*

 *You are*

 *<?php*

 *echo $\_POST[“usia”];*

 *?> years old*

*</html>*

*</body>*

Keterangan : bagian yang dicetak tebal adalah script yang semula ditulis di file *welcome.php*