

Mengganti Teks dengan PHP

Dalam membangun sebuah web, ada kalanya kita akan melibatkan penggantian text. PHP memiliki fasilitas yang sangat memudahkan kita dalam melakukan pekerjaan tersebut. Di artikel ini, kita akan membahas beberapa contoh penggantian teks.

Bagi Anda yang sering berkunjung ke website komunitas, pernahkah Anda melihat satu halaman web, penuh teks dan mengandung banyak link ke website lain? Dan, link tersebut pun bukan hanya satu, melainkan berulang-ulang. Sebagai contoh, kalau mereka menyebutkan 5 kali, maka kelima-limanya juga akan berupa link ke www.php.net.

Apakah mereka mengerjakannya sendiri secara manual? Dalam pengertian, mereka mengetikkan sendiri kode HTML `php` seperti ini setiap kali mereka ingin menyebutkan PHP? Mungkin saja. Tapi, akan terlalu merepotkan, dan sepertinya tidak praktis. Dan, yang penting, terlalu konyol untuk dilakukan.

Masalahnya, kalau hanya satu atau dua, maka tidaklah menjadi masalah. Tapi, kalau 5? Kalau 10? Dan, bagaimana pula kalau ada yang lupa? Atau, bagaimana kalau Anda menjadi terlalu lelah untuk mengusahakan hal ini? Kondisi bisa menjadi tidak konsisten, dan secara keseluruhan, hasil akhir menjadi tidak memuaskan, padahal Anda telah bekerja keras mengganti banyak kata.

Yang penting, bagaimana kalau seandainya data tersebut disimpan dalam sebuah database dan digunakan oleh banyak aplikasi? Tidak masalah kalau ditampilkan dalam web. Tapi, bagaimana kalau juga dibaca oleh client berbasis GUI yang menampilkannya dalam sebuah *text viewer* biasa? Tentu akan jadi kacau sekali.

Untuk menjaga integritas data, sebisa mungkin usahakan satu data, yang dapat diakses oleh sistem apapun juga.

Beberapa implementasi telah memungkinkan fasilitas penggantian kata otomatis. Namun, kita akan membuat sistemnya sendiri, untuk memahami bagaimana mengganti teks sendiri. Dengan demikian, pemahaman kita akan lebih baik lagi.

Setiap pembahasan di artikel ini akan menggunakan *regular expression*. Tapi, RE yang dipakai sangatlah sederhana.

Untuk melihat contoh penggantian teks di aplikasi standar Linux, cobalah perhatikan contoh berikut ini. Kita memiliki sebuah file teks dengan nama file IN dan kita akan menghasilkan sebuah file dengan nama file OUT. Berikut ini adalah isi dari file IN:

Saya mempelajari php untuk membangun website. Tak disangka-sangka, rupanya php sangatlah menyenangkan. Mari sama-sama belajar php. Kunjungilah website php sekarang juga!

Di file OUT nantinya, setiap kata `php` akan digantikan dengan `PHP`.

Berikut ini adalah contoh perintahnya:

```
$ cat IN | sed -e 's/php/<a href=http://www.php.net>PHP</a>/' > OUT
```

Penggunaan karakter `\` diperlukan karena `<` dan `/` adalah karakter spesial di shell. Berikut ini adalah isi file OUT setelah diberikan perintah tersebut:

```
Saya mempelajari <a href=http://www.php.net>PHP</a> untuk membangun website. Tak disangka-sangka, rupanya<a href=http://www.php.net>PHP</a> sangatlah menyenangkan. Mari sama-sama belajar <a href=http://www.php.net>PHP</a>. Kunjungilah website <a href=http://www.php.net>PHP</a> sekarang juga!
```

Tentu saja masih terdapat beberapa kekurangan dalam contoh tersebut. Namun, paling tidak, kita bisa melakukan penggantian teks secara sederhana. Cobalah kopikan OUT ke OUT.html dan berikan tag

pembuka dan penutup HTML, kemudian bukalah di *browser*. Semua `php` telah menjadi *hyperlink* PHP ke <http://www.php.net>.

Penulis mengenal beberapa orang yang cukup aktif di komunitas dan membuat halaman web yang secara sekilas tampak sangat sederhana, namun *engine* di bawahnya, menurut penulis adalah *engine* yang bagus. Websitenya sendiri dibuat dengan *interface* CGI menggunakan *shell script*. Memang tidak menggunakan database canggih seperti PostgreSQL, namun hampir semuanya berbasis file teks. Dengan kemampuan RE yang tinggi, manipulasi teks juga dimungkinkan. Sederhana, tampak kuno, susah dikembangkan, namun cukup untuk memenuhi kebutuhan sang pemilik. Dan, dalam konteks tertentu, itu cukup.

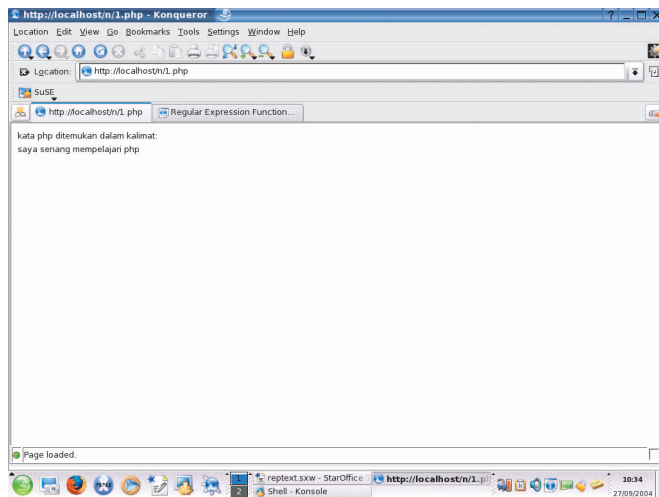
Setelah kita mendapatkan gambaran penggantian teks, kita akan memulainya dengan PHP, mulai dari contoh sangat sederhana sampai contoh yang lebih kompleks. Untuk mengikuti tutorial ini, Anda harus memiliki sebuah web server (Apache) yang mampu menjalankan script PHP.

Mencari teks

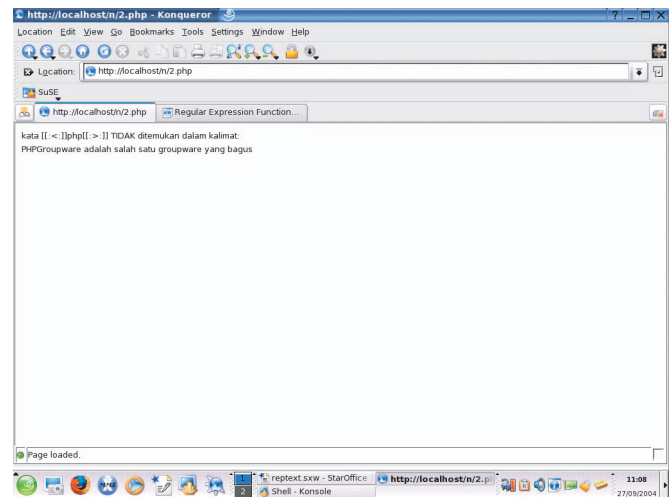
Sesungguhnya, penggantian teks tidak perlu dimulai dari pencarian teks. Anda bisa mem-bypass ini begitu saja. Namun, dengan memahami pencarian teks sederhana, kita bisa mengetahui lebih banyak contoh dan pengenalan terhadap RE. Hal ini akan sangat memudahkan penggantian teks nantinya.

Sebagai contoh, berikut ini adalah isi sebuah file `php` dengan nama `1.php`:

```
<?
$str = "saya senang mempelajari php";
$search = "php";
if (ereg($search, $str))
```



Contoh output 1.php.



Contoh output 2.php.

```

{
    echo nl2br("kata
$search ditemukan dalam kalimat:
\n$str");
} else
{
    echo nl2br("kata
$search TIDAK ditemukan dalam
kalimat: \n$str");
}
?>

```

Kita menggunakan fungsi `ereg(search, string)` untuk mencari `search` di dalam `string`. Fungsi ini akan mengembalikan nilai benar apabila pencarian ditemukan. Namun, harap diperhatikan, fungsi ini mencari secara *case sensitive*. Membedakan huruf besar dan huruf kecil. Cobalah mengganti `$search` menjadi `PHP`, dan menurut 1.php, kata `PHP` tersebut tidak ditemukan.

Hal ini benar bagi sistem, namun akan susah dipahami oleh manusia. Apalagi pengunjung *website* Anda. `PHP`, `php`, `PhP` dan sebagainya adalah sama dengan `php` bagi pengunjung. Oleh karena itu, pencarian tidak boleh dilakukan secara *case sensitive*. Harus *case-insensitive*.

Caranya mudah sekali. Cukup ganti fungsi `ereg()` dengan `eregi()`. Parameternya sama. Anda bisa mencobanya.

Sekarang, kopikanlah 1.php ke file 2.php, dan simpanlah 1.php. Kita akan membuat contoh yang lebih kompleks. Berikut ini adalah isi 2.php:

```

<?
    $str = "PHPGroupware
adalah salah satu groupware yang
bagus";
    $search = "php";
    if (eregi($search,
$str))
    {
        echo nl2br("kata
$search ditemukan dalam kalimat:
\n$str");
    } else
    {
        echo nl2br("kata
$search TIDAK ditemukan dalam
kalimat: \n$str");
    }
?>

```

Menurut Anda, apakah yang akan terjadi apabila kita menjalankan 2.php tersebut? Apakah kata `php` dianggap ketemu? Jawabannya adalah Ya. Menilik karakternya, tentu saja `php` ditemukan. Tapi, bukan ini yang

kita inginkan. Di pembahasan berikutnya, Apabila kita mengganti setiap `php` yang ditemukan dengan `PHP`, maka jadinya akan sangat lucu dan tidak sesuai keinginan.

Kenapa? Karena kita menginginkan `PHPGroupware` lah yang akan dibuat *hyperlink* ke *website* project ini. Bukan bagian dari `PHPGroupware`, yaitu `PHP` yang akan dijadikan link ke `http://www.php.net`. Oleh karena itu, pencarian tersebut harus gagal. Kita hanya boleh menganggap `PHP` ketemu apabila merupakan suatu kata sendiri, bukan bagian dari suatu kata.

Caranya, gunakan aturan RE berikut. Gantilah `$search` Anda menjadi:

```
$search = "[[:<:]]php[[:>:]]";
```

Penggunaan RE tersebut pada fungsi `eregi()` akan membuat pencarian hanya akan berhasil apabila `PHP` merupakan suatu kata tersendiri. Bukan bagian dari suatu kata. Berupa kata sendiri artinya pencarian `php` dalam `'python,php,perl'` akan dianggap ketemu. Namun, pencarian `php` dalam `PHPGroupware` akan dianggap tidak ketemu.

Sampai di sini, kita sudah bisa mencari keberadaan satu kata dalam suatu kalimat. Ini adalah salah satu dasar penting dalam penggantian teks dalam suatu *website*. Jangan sampai kita mengganti hanya sebagian karakter dalam satu kata.

Mengganti teks

Setelah kita berhasil mencari teks, kita akan melanjutkan ke penggantian teks seder-

hana. Apabila dalam pencarian teks kita menggunakan fungsi `ereg()` dan `eregi()`, maka dalam penggantian teks, kita akan menggunakan fungsi `ereg_replace()` atau `eregi_replace()`. Berikut adalah deskripsi fungsinya:

```
string ereg_replace ( string
pattern, string replacement,
string string)
string eregi_replace ( string
pattern, string replacement,
string string)
```

Berikut adalah contoh sederhana penggantian teks dalam script 3.php:

```
<?
<
    $str = "Saya senang
belajar PHP";

    $pattern =
"[[<:]]php[[>:]]";
    $replace = "<b>PHP</b>";

    echo eregi_
```

```
replace($pattern, $replace,
$str);
```

```
?>
```

Contoh tersebut akan mengganti setiap kata PHP dengan PHP yang dibelkan. Penggantian hanya terjadi pada kata PHP (bagaimanapun PHP ditulis) karena penggunaan `$pattern = "[[<:]]php[[>:]]"`.

Tentu saja, kita dengan mudah dapat mengganti PHP tersebut menjadi `PHP`. Ganti saja variabel `$replace` menjadi:

```
$replace = "<a href=http://www.
php.net>PHP</a>";
```

Dengan bekal ini, kita bisa memulai membuat engine sederhana untuk mengganti setiap kata yang kita inginkan dengan kata ditambah tag tertentu sehingga menjadi hyperlink ke website tertentu misalnya.

Penerapannya sangatlah banyak. Sebagai contoh, ketika Anda membuat sebuah hala-

man yang berisikan banyak project seperti disebutkan pada awal tulisan. Atau, ketika Anda membuat search engine sederhana yang akan menebalkan teks yang dicari oleh user dalam hasil pencarian.

Fasilitas seperti ini, akan sangat membantu user dan secara tidak langsung, akan meningkatkan value website Anda. Apalagi jika digunakan secara konsisten.

Permasalahan berikut adalah, contoh-contoh sebelumnya masih belum merepresentasikan keadaan di dunia nyata. Apa yang kita ganti masih berupa satu kata. Bagaimana kalau kita ingin mengganti beberapa kata dalam menampilkan suatu konten? Perhatikanlah contoh berikut ini.

Contoh kasus

Contoh berikut ini adalah script PHP yang akan membaca sejumlah kata yang ingin diganti dan langsung mengganti semua kata tersebut. Sejumlah kata yang ingin diganti akan disimpan ke dalam sebuah file terpisah untuk lebih mudahnya. Demikian juga dengan teks aslinya.

Program Intensif LINUX Profesional

LINUX Server Development Mampu membangun server & mengelola jaringan komputer berbasis Linux & penerapan aplikasinya pada internet / intranet beserta system security-nya.

MATA KULIAH DASAR :

- **Pengantar Komputer**
Komponen & Konfigurasi Komputer & Jaringan Komputer / LAN
- **Linux Fundamental**
(Basic User, X-window, System Administrator, Networking)
- **Internet & Design WEB**
Browser, Search Engine, Email, FTP, HTML, CSS & JavaScript

MATA KULIAH INTI :

- **Shell Programming**
- **Advanced System Administration**
- **Security & Advanced Networking**
- **PHP & MySQL**

WAKTU KULIAH:

- ❖ **Sabtu s.d. Minggu**
Jam: 08.00 s.d. 12.00 WIB.
Jam: 14.00 s.d. 18.00 WIB.
- ❖ **Sabtu - Minggu**
Jam: 08.00 s.d. 12.00 WIB.
Jam: 14.00 s.d. 18.00 WIB.

BEBAS BIAYA
PENDAFTARAN

**&
*Discount 10%**

* Untuk Pembayaran
Biaya Kuliah di Muka
(Lunas)
Berlaku
Desember '04
s.d.
Januari '05

FASILITAS:

- ☉ Ruang Kuliah Full AC
- ☉ Min PIII + Networking
- ☉ RAM 128 + Multimedia
- ☉ CD Linux
- ☉ Disket
- ☉ Sertifikat
- ☉ Internet
- ☉ Modul Belajar
- ☉ Block Notes
- ☉ T-Shirt
- ☉ Cutten Bag



LEMBAGA PENDIDIKAN KOMPUTER NURUL FIKRI
Jl. Margonda Raya No.522 Depok 16424 ☎/Fax. (021) 7874223, 7874224, 77206991
<http://www.nurulfikri.com> | email: info@nurulfikri.com

Branch Office

Jl. Mampang Prapatan X/4 Jakarta 12790
☎ (021) 7975235 Fax: 7947115, 7901205

Sentra Niaga Blok B.I/12
Jl. Ahmad Yani - Bekasi ☎ (021) 8853537

BERKUALITAS
TERPERCAYA

Pelopor Training LINUX di Indonesia

PAKET
Awal & Akhir Tahun



File rep.php – Definisi kata yang akan diganti:

```
<?
$REP = array (
    "php" => "<a href=http://www.php.net>PHP</a>",
    "python" => "<a href=http://www.python.org>PYTHON</a>",
    "GTK" => "<a href=http://www.gtk.org>GTK</a>",
    "Linux" => "<a href=http://www.linux.org>LINUX</a>",
    "GNU" => "<a href=http://www.gnu.org>GNU</a>",
    "Infolinux" => "<a href=http://www.infolinux.web.id>INFOLINUX</a>",
);
?>
```

Anda bisa menambahkan kata sendiri dengan mengikuti aturan *array* PHP. Kita menggunakan *array* untuk mempermudah *data processing*. Mungkin bagi Anda yang terbiasa dengan memisahkan kata (*split*), akan terasa lebih nyaman apabila menggunakan file teks (misal PHP=http://www.php.net). Pendekatan dengan XML-based

source juga dimungkinkan. Semua didukung oleh PHP.

File 4.txt – File data berisikan teks asli: *Infolinux adalah majalah bulanan Linux yang membahas berbagai hal.*

Mulai dari sistem operasi itu sendiri, administrasinya serta berbagai pembahasan pemrograman.

Contoh pembahasan pemrograman mencakup php, python, GTK dan lain sebagainya. Berbagai free software dari proyek gnu tentunya juga dibahas.

File 4.php – Prosesor:

```
<?
include ("rep.php");
$f = fopen("4.txt", "r");
while (!feof($f))
{
    $buffer =
fgets($f, 4096);
    $temp = $buffer;
    foreach ($REP as
    $k => $v)
    {
        $temp =
ereg_replace("[:<:]" . $k .
"[:>:]", $v, $temp);
    }
    echo
nl2br($temp);
}
?>
```

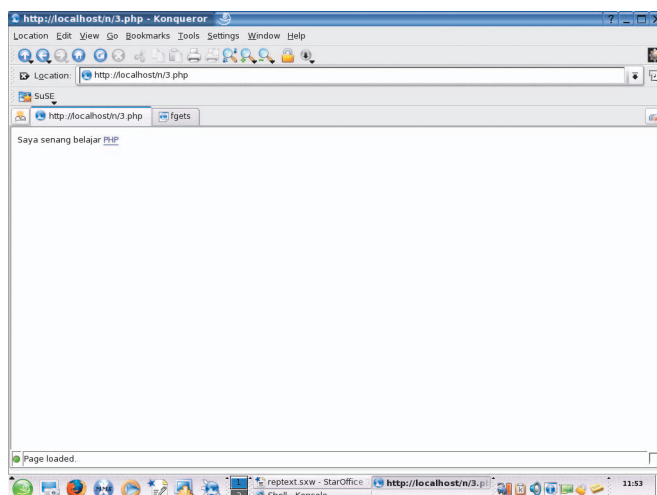
Penjelasan kode:

- Mula-mula, kita menginclude rep.php.
- Kemudian, kita membuka file 4.txt.
- Setelah itu, dalam perulangan sampai habis membaca 4.txt, kita membaca baris demi baris.
- Untuk setiap baris yang kita dapatkan, kita akan membaca setiap entri dari array \$REP dan melakukan pergantian teks. Pergantian dilakukan per kata.
- Kita mencetak hasilnya ke browser. Agar sama dengan file aslinya, kita menggunakan bantuan fungsi nl2br(). Fungsi ini akan mengganti new line dengan
.

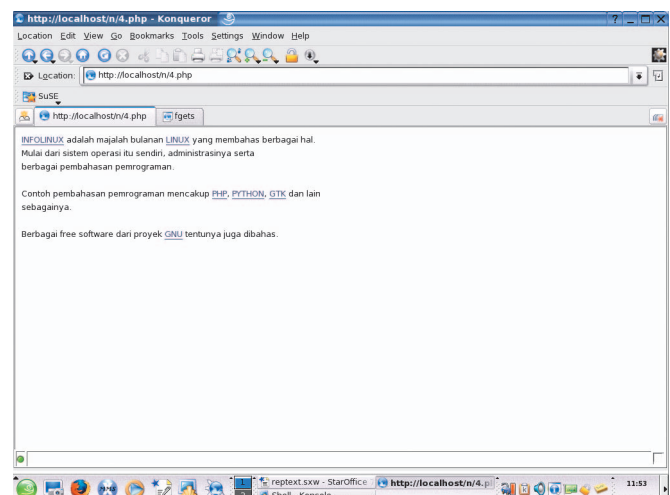
Demikianlah contoh penggantian teks secara lengkap. Sumber data tidak harus selalu dari file teks. Anda juga bisa menggunakan database postgresql misalnya.

Program ini tentu saja memiliki kekurangan. Masalah utama adalah penggantian teks yang sangat serakah. Anda bisa bayangkan kalau entri \$REP telah menjadi begitu panjang? Untuk mengganti satu baris teks (kalaupun ada yang perlu diganti dan ini adalah salah satu kelemahan yang serius.), pencarian ke dalam keseluruhan \$REP akan dilakukan. Pencarian ini akan memakan waktu. Regular expression bukanlah hal yang sepele dan sangat membutuhkan resource yang cukup besar.

Solusi lain adalah dengan memecah \$REP. Anda bisa menggunakan cara lain yang lebih efisien dalam mengganti teks, apabila Anda ingin membuatnya sendiri. **Noprianto** (noprianto@infolinux.co.id)



Contoh output 3.php.



Contoh output 4.php.