Bab 5

Penutup

Regex bukan Hammer

Ada pepatah yang sering sekali dipakai orang komputer, "when the only tool you have is a hammer, everything looks like a nail." Artinya, kalau kita hanya memiliki sebuah alat (atau sedang tergila-gila pada alat tersebut), maka kita jadi mencoba menyelesaikan semua persoalan menggunakan alat tersebut. Padahal, belum tentu alat tersebut yang paling tepat —atau tepat sama sekali!— untuk persoalan yang dimaksud. Kalau boleh saya pakai analogi yang lebih modern di masyarakat, karena orang saat ini tergila-gila dengan ponsel, maka segala sesuatu dilakukan dengan ponsel. Mengirim kartu ucapan dengan SMS, padahal dengan kartu pos biasa bisa lebih personal dan berkesan (sehingga lebih efektif). Bahkan ada yang menceraikan istri lewat SMS. Sesuatu yang rasanya kurang manusiawi.
Regex memang ampuh dan asyik dalam mengolah teks. Tapi tidak harus langsung melirik regex untuk menyelesaikan persoalan memroses teks. Beberapa persoalan berikut misalnya, sebaiknya menggunakan cara lain dulu sebelum menggunakan regex.

Memroses HTML dan XML secara kompleks. Misalnya memanipulasi atribut (menambah, mengubah, menghapus atribut), memvalidasi HTML, merapikan (*pretty-printing*), dan sebagainya. Operasi-operasi seperti ini sebaiknya menggunakan parser HTML atau XML khusus, karena keduanya bukanlah teks linear melainkan teks berstruktur. Rata-rata bahasa pemrograman menyediakan library untuk ini. Tapi untuk operasi-operasi yang *quick-n-dirty*, seperti pada contoh resep 5-8, regex cukup enak digunakan.

Melakukan validasi terhadap data-data yang umum. Misalnya mengecek sintaks alamat e-mail dan URL, parsing string tanggal atau waktu, mengecek nama file yang sah pada sebuah OS tertentu, dan sebagainya. Untuk pekerjaan seperti ini biasanya telah tersedia fungsi atau library pada bahasa pemrograman favorit Anda. Lagipula, aturan untuk alamat e-mail dan URL sebetulnya kompleks dan regex buatan Anda belum tentu telah benar-benar teruji kebenarannya. Sementara untuk data seperti tanggal dan waktu ada begitu banyak variasi notasi, sehingga sebaiknya menggunakan library yang telah berisi koleksi relatif lengkap berbagai notasi tersebut.

Melakukan tugas-tugas pemrosesan teks umum lainnya. Misalnya memparsing file CSV. Untuk hal-hal seperti ini juga umumnya telah tersedia library. *Don't reinvent the wheel*, kecuali memang Anda sedang membuat regex dalam rangka latihan.

Melakukan tugas-tugas yang sederhana sehingga operasi string biasa juga cukup. Misalnya, mengecek apakah karakter pertama adalah spasi. Atau, memastikan

bahwa string tidak kosong. Di bahasa-bahasa tertentu seperti Perl memang regex digunakan untuk hampir segala macam pekerjaan, termasuk hal-hal sederhana seperti ini. Ini karena, selain budaya bahasa itu sendiri, regex di Perl telah dioptimasi sehingga operasi sederhana seperti ini secara internal bisa diubah menjadi operasi string biasa. Di bahasa-bahasa lain yang tidak terlalu "gila regex", umumnya yang dianjurkan adalah menggunakan fungsi/metode string biasa. Misalnya, di Python ada metode startswith(), endswith(), isdigit(), isspace(), bahkan hingga expandtabs().

Beberapa Tips Membuat Regex

i biasanya lambat. Mode case-insensitive biasanya lebih lambat daripada case-sensitive, karena untuk setiap karakter mesin regex juga perlu mencocokkannya dengan casing yang berbeda ("a" dengan "a" dan juga "A", dan sebagainya). Namun pada mayoritas kasus, hal ini tidak perlu Anda khawatirkan. Menggunakan i memang nyaman terutama untuk HTML, karena tag-tag HTML dapat ditulis dengan campuran huruf besar dan kecil (<title>, <TITLE>, <Title>).

Hati-hati dengan copy-paste. Saya biasanya sering melakukan copy-paste terutama URL ke dalam regex. Biasanya kesalahan yang sering terjadi adalah tidak meng-escape ? atau *. Contohnya, m#http://www.foo.bar/test.php?query=123# di mana URL tersebut hasil copy-paste. Anda akan bingung mengapa pola regex tidak cocok dengan string. Penyebabnya adalah ? harus di-escape menjadi \?. Demi akurasi pola regex, pertimbangkan juga untuk mengubah . menjadi \., meskipun ini tidak wajib.

Hati-hati dengan substitusi variabel. Jika Anda menggunakan m// di Perl atau preg_match("...") di PHP dengan kutip ganda, berhati-hati. Karena "$FOO" akan diinterpolasi dengan nilai variabel $FOO—hal ini bisa jadi memang Anda inginkan dan bisa jadi tidak. Tentu saja, hal ini hanya perlu dikuatirkan jika pola regex Anda mengandung $ di tengah-tengah pola.

Mode x dan spasi. Mode x memang dianjurkan untuk membuat pola mudah dibaca, tapi kadang-kadang kita lupa bahwa spasi harus selalu ditulis dengan \s pada mode ini.

Greedy dan non-greedy. Mode greedy biasanya lebih berbahaya, misalnya /<!—.*—>/s dapat menelan seluruh isi HTML Anda karena <!— cocok dengan awal komentar pertama dan —> cocok dengan akhir komentar *terakhir*. Setiap kali menggunakan .+ atau .* dan mode s, selalu ingatlah untuk bertanya apakah mode greedy atau non-greedy yang Anda inginkan. Biasanya (tapi tidak selalu) mode non-greedy yang Anda perlukan.

Pola kompleks dan kecepatan. Kehadiran banyaknya .* dan .+ biasanya dapat amat memperlambat proses pencocokan. Sebisa mungkin, sebaiknya Anda menggunakan penjangkaran (/^...$, /^.../, /...$/) atau atomic group, (?>...), untuk membantu mesin regex mengurangi jumlah backtracking yang harus dilakukan.

Lindungi data. Jika Anda menggunakan substitusi regex untuk merename file, memroses isi file, atau mengubah-ubah data di database; jangan lupa untuk selalu melakukan backup dulu sebelum mencoba-coba. Jangan sampai kesalahan menulis pola regex berakibat Anda kehilangan data penting.

Diskusi!

Saya membuat milis id-regex@yahoogroups.com yang bisa Anda pakai untuk berdiskusi seputar masalah regex, tak peduli bahasa apa yang Anda gunakan (PHP/Perl, Python, Java, atau bahkan ASP/VB/.NET).

Akhir kata, sama seperti semua bahasa lain, kunci menguasainya adalah dengan latihan. Mulailah menggunakan regex dalam program-program Anda; gunakanlah **grep**, **awk**, **sed**, atau **perl** di shell Unix; bacalah modul-modul Perl, Ruby, atau paket-paket PEAR atau aplikasi PHP di mana terdapat penggunaan pola-pola regex. Moga-moga regex dapat digunakan untuk mempermudah hidup Anda!