

Andre Pratama

HTML

Uncover

Panduan Belajar HTML untuk Pemula

DuniaIlkom

HTML Uncover

Panduan Belajar HTML untuk Pemula

DuniaIlkom

Buku ini dijual di <http://leanpub.com/htmluncover>

Versi ini diterbitkan pada 2018-01-08



Ini adalah sebuah buku [Leanpub](#). Leanpub memberdayakan penulis dan penerbit dengan proses Lean Publishing. [Lean Publishing](#) adalah model penerbitan ebook dalam-proses menggunakan piranti ringan dan sejumlah iterasi untuk memperoleh masukan dari pembaca, menerapkan pivot hingga Anda dapat mewujudkan komposisi buku yang pas dan menarik.

© 2015 - 2018 DuniaIlkom

Contents

Ucapan Terimakasih	i
Tentang Penulis	ii
Kata Pengantar	iii
Asumsi / Pengetahuan Dasar	v
Contoh Kode Program	vi
1. Berkenalan Dengan HTML	1
1.1 Pengertian HTML	1
1.2 Fungsi HTML	2
1.3 Melihat Sekilas HTML	3
1.4 Mengenal Jenis-jenis Bahasa Pemrograman Web	5
2. Sejarah dan Perkembangan HTML	7
2.1 Sejarah HTML	7
2.2 Teknologi dan fitur baru dalam HTML5	15
2.3 Apa Yang Harus Saya Kuasai?	16
3. Web Browser dan Text Editor	17
3.1 Mempersiapkan Web Browser	17
3.2 Memilih Text Editor	21
3.3 Menjalankan File HTML Pertama Anda	23
3.4 Teks Editor Alternatif: Komodo Edit	26
3.5 Teks Editor Alternatif: Atom	28
3.6 Bagaimana dengan Sublime Text?	30
4. Struktur Dasar HTML	32
4.1 Pengertian Tag, Atribut dan Element HTML	32
4.2 Struktur Dasar HTML	34
4.3 Mengenal HTML Tree / Document Object Model (DOM)	37
4.4 Aturan Dasar Penulisan HTML	38
4.5 Standar & Validasi HTML	44
4.6 Block Level Element dan Inline Level Element	47
5. Head Element	50

CONTENTS

5.1	Meta Element	50
5.2	Link Element	55
5.3	Script Element	58
5.4	Style Element	58
5.5	Title Element	59
5.6	HTML5Shiv	60
5.7	Penutup: Head Element	64
6.	Text Formatting Element	67
6.1	Paragraf Element	67
6.2	Anchor Element (a)	70
6.3	Heading Element (h1 – h6)	77
6.4	Emphasis, Strong, Bold, dan Italic Element	79
6.5	Underline dan Strikethrough Element	80
6.6	Superscript dan Subscript Element	82
6.7	Preformatted Element	83
6.8	Code, Samp, KBD, dan Var Element	84
6.9	Cite, Quote dan Blockquote Element	85
6.10	Mark Element	86
6.11	Ruby Element	87
6.12	Abbr Element	89
6.13	Address Element	90
6.14	Bdo Element	91
6.15	Dfn Element	92
6.16	Small Element	93
6.17	Wbr Element	94
6.18	Deprecated Element dan Atribut	97
6.19	HTML Entity	99
7.	List Element	103
7.1	Ordered List	103
7.2	Unordered List	108
7.3	Description List	113
8.	Image Related Element	115
8.1	Image Element	115
8.2	Map Element, Area Element, dan Atribut Usemap	132
8.3	Figure Element dan Figcaption Element	136
9.	Audio dan Video Element	141
9.1	Audio Element	141
9.2	Atribut Src	141
9.3	Atribut Controls	142
9.4	Atribut Autoplay	143
9.5	Atribut Loop	143
9.6	Atribut Muted	144

CONTENTS

9.7	Atribut Preload	144
9.8	Mengenal Audio Format	146
9.9	Perbedaan Antara Format Audio Dengan File Extension	148
9.10	Dukungan Format Audio oleh Web Browser	148
9.11	Menggunakan Beberapa Format File Audio	149
9.12	Pemberitahuan untuk Web Browser tua	150
9.13	Video Element	151
9.14	Atribut Autoplay	152
9.15	Atribut Loop	152
9.16	Atribut Muted	152
9.17	Atribut Preload	153
9.18	Atribut Width dan Height	154
9.19	Atribut Poster	154
9.20	Mengenal Video Format	155
9.21	Mengenal Standar Video	157
9.22	Dukungan Format Video oleh Web Browser	158
9.23	Menggunakan Beberapa Format File Video	158
9.24	Pemberitahuan untuk Web Browser Tua	159
10.	Table Element	161
10.1	Struktur Dasar Tabel (table, td dan tr element)	161
10.2	Atribut Border	163
10.3	Atribut Rules	166
10.4	Atribut Cellspacing	169
10.5	Atribut Cellpadding	172
10.6	Atribut Bgcolor	175
10.7	Atribut Width dan Height	178
10.8	Atribut Align	185
10.9	Atribut Valign	189
10.10	Atribut Rowspan dan Colspan	191
10.11	Th Element	194
10.12	Caption Element	195
10.13	Colgroup dan Col Element	196
10.14	Thead, Tbody dan Tfoot Element	199
10.15	Mengisi Sel Tabel	200
10.16	Membuat Layout Menggunakan Tabel	202
10.17	Penutup: Table Element	204
11.	Form Element	208
11.1	Form Element	208
11.2	Input Element Type Text (text box)	210
11.3	Input Element Type Submit	217
11.4	Input Element Type Reset	221
11.5	Input Element Type Password	222
11.6	Input Element Type Checkbox	223
11.7	Input Element Type Radio (radio button)	225

CONTENTS

11.8	Textarea Element	228
11.9	Select Element dan Option Element (menu dropdown)	232
11.10	Optgroup Element	238
11.11	Input Element Type File	239
11.12	Input Element Type Hidden	242
11.13	Input Element Type Image	243
11.14	Input Element Type Button	245
11.15	Label Element	247
11.16	Fieldset Element dan Legend Element	250
11.17	Button Element	252
11.18	Penutup: Form Element	252
12.	HTML5 Form Element	258
12.1	Validasi Form	258
12.2	Input Element Type Color	258
12.3	Input Element Type Email	260
12.4	Input Element Type URL	262
12.5	Input Element Type Search	263
12.6	Input Element Type Number	264
12.7	Input Element Type Range	268
12.8	Input Element Type Tel	269
12.9	Input Element Type Date, Datetime, Datetime-local, Time, Month, dan Week	270
12.10	Atribut Autofocus	274
12.11	Atribut Placeholder	275
12.12	Atribut Autocomplete	276
12.13	Atribut Required	277
12.14	Atribut Form	278
12.15	Datalist Element dan Atribut List	279
12.16	Atribut Pattern	281
12.17	Atribut Novalidate	283
12.18	Atribut Formaction, Formenctype, Formmethod dan Formtarget	283
12.19	Progress Element	285
12.20	Meter Element	286
12.21	Output Element	287
12.22	Keygen Element	288
12.23	Atribut Contenteditable	289
12.24	Atribut Designmode	290
13.	Embed, Object dan Iframe Element	293
13.1	Embed Element	293
13.2	Object Element	294
13.3	Iframe Element	296
13.4	Pilih Embed, Object, atau iFrame?	301
14.	HTML5 Canvas Element	302
14.1	Canvas Element	302

CONTENTS

14.2	Berkenalan dengan JavaScript untuk Canvas	303
14.3	Membuat Garis dalam Canvas	305
14.4	Mengisi Warna Canvas	309
14.5	Shape untuk Canvas	312
14.6	Membuat Teks di dalam Canvas	315
14.7	Penutup: HTML5 Canvas	316
15.	Scalable Vector Graphics (SVG)	318
15.1	Pengertian SVG	318
15.2	Raster Image vs Vector Image	318
15.3	SVG Element	319
15.4	Membuat Garis dengan SVG	321
15.5	Membuat Persegi dengan SVG	325
15.6	Membuat Lingkaran dan Elips dengan SVG	326
15.7	Membuat Polygon dengan SVG	328
15.8	Membuat Garis Bersambung dengan SVG	329
15.9	Membuat Teks dengan SVG	330
15.10	Menggunakan External SVG	331
16.	HTML5 Semantic Tag (Layout)	334
16.1	Mengenal Struktur Halaman Web	334
16.2	Membuat Layout dengan Div Element	335
16.3	Pengertian Semantic Tag / Semantic Element HTML5	338
16.4	Header Element	339
16.5	Section Element	340
16.6	Article Element	340
16.7	Nav Element	342
16.8	Aside Element	342
16.9	Main Element	343
16.10	Footer Element	343
16.11	Hgroup Element	344
16.12	Contoh Layout Halaman HTML5	344
17.	HTML 5 Update	350
17.1	Update HTML 5.1	350
17.2	Picture Element	351
17.3	Details dan Summary Element	356
17.4	Menu dan MenuItem Element	359
17.5	Update HTML 5.2	361
17.6	Dialog Element	361
17.7	Update HTML 5.3	364
Kursus Kilat JavaScript Dasar	365	
Apa itu JavaScript?	365	
Cara Menggunakan JavaScript	365	
Menggunakan tag <noscript>	369	

CONTENTS

Sekilas Event JavaScript	370
Kursus Kilat CSS Dasar	373
Apa itu CSS?	373
Cara Menggunakan CSS	373
Pengertian Selector, Property dan Value	374
Contoh penggunaan Internal CSS	374
Contoh penggunaan External CSS	377
Contoh penggunaan Inline CSS	379
Sekilas CSS Property	380
Penutup: HTML Uncover	383
Referensi HTML	384
Daftar Pustaka	385

Ucapan Terimakasih

Dalam kesempatan ini saya ingin mengucapkan terimakasih kepada Allah Subhanahu wa Ta'ala karena dengan karuniaNya saya masih diberi kesempatan dan kesehatan untuk bisa menulis buku ini.

Selanjutnya kepada keluarga saya yang terus memberi motivasi dan dukungan tiada henti untuk terus mengembangkan duniaIlkom.

Terakhir kepada rekan-rekan pembaca dan pengunjung setia DuniaIlkom yang terus memberi respon positif agar saya terus berkarya. Terutama kepada donatur dan rekan-rekan yang telah membeli buku ini. Terimakasih :)

Padang Panjang, 2018

Penulis

Andre Pratama

www.duniaIlkom.com

Tentang Penulis

Andre Pratama



Saat ini memilih karir sebagai praktisi dan penulis di duniailkom.com. Menamatkan kuliah S1 Ilmu Komputer di Universitas Sumatera Utara pada tahun 2010. Sempat terjun ke dunia kerja melalui ODP (*Officer Development Program*) di Bank BUMN terbesar di Indonesia. Di sela-sela kesibukan, mendirikan situs duniailkom.com pada tahun 2012.

Karena kecintaan akan menulis dan programming, akhirnya memutuskan untuk keluar dari dunia perbankan dan fokus mengelola duniailkom di akhir tahun 2014. Berusaha untuk menjadikan duniailkom sebagai salah satu media belajar programming dan ilmu komputer terbaik di Indonesia.

Berdomisili di kota Padang Panjang, Sumatera Barat, Andre bisa dihubungi melalui email duniailkom di duniailkom@gmail.com, facebook: facebook.com/belajar.duniailkom¹, dan twitter: twitter.com/duniailkom².

¹<https://www.facebook.com/belajar.duniailkom>

²<https://twitter.com/duniailkom>

Kata Pengantar

Versi pertama **HTML Uncover** saya rilis akhir Mei 2015. Buku pertama, sekaligus menjadi tumpuan harapan dari duniaIlkom. Dengan bahasa yang mudah dipahami serta kelengkapan materi, saya berharap buku ini bisa menjadi salah satu media belajar terbaik untuk menguasai HTML.

Tidak terasa sudah hampir 3 tahun berlalu. Saat ini eBook **HTML Uncover** sudah dibaca oleh lebih dari 3200 rekan-rekan duniaIlkom. Ini merupakan suatu bentuk dukungan, kepercayaan, dan pencapaian yang luar biasa bagi saya pribadi.

Merilis sebuah eBook memiliki resiko yang sangat besar (mudah di copy-paste), tapi saya memilih cara ini agar harga buku tidak terlalu mahal. Jika buku ini di cetak, biayanya sudah mencapai ratusan ribu (di luar ongkos kirim), apalagi mengingat ketebalan buku ini. Versi eBook juga memudahkan bagi teman-teman di daerah agar bisa membeli buku tanpa harus memikirkan ongkos kirim.

Walaupun terdapat beberapa pihak yang kurang bisa menghargai (dengan menggandakan eBook ini), saya akan terus berusaha untuk menghadirkan lebih banyak eBook web programming. Dukungan dari rekan-rekan yang telah membeli versi resminya ke duniaIlkom akan selalu menjadi motivasi saya untuk terus berkarya, terimakasih!

Sebagai bentuk dedikasi dan konsistensi, sudah saatnya bagi saya untuk men-update materi di dalam eBook **HTML Uncover**. Sebagian besar materi tetap relevan dan masih sesuai dengan perkembangan terbaru HTML5. Namun ada beberapa bagian yang perlu diperbarui, termasuk cover buku ini yang kini tampil lebih fresh.

Buku **HTML Uncover** saya tujuhan bagi pembaca pemula yang ingin mempelajari HTML. Saya berusaha menyajikan bahasan yang padat, detail namun dengan bahasa yang mudah dipahami. Kita akan mulai dari pengertian HTML, perkembangan HTML, menjalankan file HTML hingga materi-materi terbaru HTML5 seperti *canvas*, *audio*, *video*, dan *form validation*.

Pembahasan dalam buku saya bagi dalam beberapa bab serta 2 bab tambahan pengantar CSS dan JavaScript. Bab-bab dikelompokkan sesuai dengan materi yang dibahas. Jika anda baru mengenal HTML dan web programming, saya sarankan untuk membaca secara bertahap dari bab pertama hingga terakhir.

Materi dalam buku ini saya rancang sesederhana mungkin. Jika merasa kesulitan, ada pertanyaan atau saran perbaikan materi, bisa menghubungi saya via email ke duniaIlkom@gmail.com.

Semoga buku HTML Uncover ini bisa menjadi buku pengantar terbaik dalam langkah anda menjadi seorang web programmer. Sampai jumpa di bab terakhir :)

Terimakasih untuk tidak memperbanyak / mengedarkan / meng-copy eBook ini

Menulis sebuah buku hingga ratusan halaman butuh waktu yang tidak sebentar. Belum lagi saya harus berjuang mempelajari referensi yang kebanyakan dalam bahasa inggris. Ini saya lakukan agar pembaca bisa mendapatkan materi yang detail, update, dan berkualitas.

Saya menyadari kekurangan sebuah ebook adalah mudah dicopy-paste dan disebarluaskan. Tapi dengan eBook, harga buku bisa ditekan. Selain tidak perlu mencetak, eBook DuniaIlkom ini bisa di dapat dengan murah, termasuk bagi teman-teman di daerah yang ongkos kirimnya lumayan mahal (jika berbentuk buku fisik).

Atas dasar itulah saya mohon kerjasamanya dari rekan-rekan semua **untuk tidak memperbanyak, menggandakan, atau mengupload ulang buku ini di forum, situs dan media lain**.

Saya juga berharap rekan-rekan tidak memposting materi apapun yang ada di dalam buku ini. Jika ingin sebagai bahan artikel untuk postingan blog/situs, silahkan ambil materi yang ada di website duniaIlkom (jangan yang dari buku).

Apabila rekan-rekan memperoleh buku ini **bukan** dari DuniaIlkom, saya mohon bantuan donasinya untuk membeli versi asli. Donasi pembelian buku ini adalah sumber mata pencarian saya untuk menafkahi keluarga. Lisensi atau hak guna buku ini hanya untuk 1 orang, yakni yang telah membeli langsung ke **duniaIlkom@gmail.com**.

Dengan kualitas yang ditawarkan, harga buku ini cukup terjangkau. Buku ini saya buat dengan waktu yang tidak sebentar, hingga berbulan-bulan, dan kadang sampai tengah malam. Bantuan donasi dari rekan-rekan yang membeli buku secara resmi sangat saya hargai, selain mendapat ilmu yang berkah, ini juga bisa menjadi penyemangat saya untuk terus berkarya dan menghadirkan ebook-ebook programming berkualitas lainnya.

Untuk yang membeli dari DuniaIlkom, saya ucapan banyak terimakasih :)

Anda diperbolehkan untuk:

- Mencetak eBook ini untuk keperluan pribadi dan dibaca sendiri.
- Mencopy eBook ini ke laptop/smartphone/laptop milik sendiri.
- Membuat ringkasan buku untuk digunakan sebagai bahan ajar (bukan keseluruan isi buku).

Anda tidak dibolehkan untuk:

- Mencetak eBook ini untuk dibaca oleh orang lain, walaupun gratis.
- Mencopy eBook ini untuk dijual ulang, maupun dibagikan kepada orang lain dengan gratis.
- Membeli buku ini untuk dibaca bersama-sama (lisensi buku ini hanya untuk 1 orang).
- Mengambil sebagian atau seluruh isi buku untuk di publish ke blog, situs, artikel, dan media publik lain.
- Membagikan eBook ini kepada murid/siswa/mahasiswa (jika digunakan untuk bahan pengajaran).

Asumsi / Pengetahuan Dasar

Buku **HTML Uncover** ini ditujukan untuk pemula yang baru mulai belajar web programming. Tidak ada skill khusus yang disyaratkan, saya akan membahas semuanya dari dasar, mulai dari cara menginstall teks editor, cara menjalankan file HTML, hingga pembahasan lanjutan seperti form validation HTML5, canvas, dan SVG.

Namun saya berasumsi anda sudah cukup familiar menggunakan web browser seperti **Google Chrome** atau **Mozilla Firefox**.

Jika belum pernah mempelajari kode HTML sebelumnya, saya sarankan untuk mengikuti bab dalam buku ini secara berurutan, mulai dari bab 1 hingga akhir. Pembahasan dalam sebuah bab bisa saja menggunakan materi dari bab sebelumnya.

Tips Menghapal Kode HTML

Ketika mempelajari buku ini, anda mungkin akan bertanya, bagaimana cara saya menghapal kode-kode HTML ini?

Tidak perlu dihapal! Cukup pahami saja cara penggunaannya. Jadikan buku **HTML Uncover** sebagai referensi jika nanti mulai terjun ke pembuatan program yang sebenarnya.

Tidak ada programmer yang hapal seluruh kode HTML. Tapi jika sudah paham cara penggunaannya, akan sangat mudah memahami kembali kode-kode ini (dengan melihatnya sekilas).

Sebagai contoh analogi, saya yakin sebagian besar dari kita pernah belajar **geometri**. Masih ingatkah dengan rumus menghitung keliling lingkaran?

Jika anda seperti saya (yang sudah puluhan tahun tamat SD), mustahil hapal rumus ini. Tapi jika ada yang nanya, saya tinggal *googling* sebentar, dan rumusnya adalah $2\pi r$. Saya tidak perlu membaca lebih jauh, karena sudah paham apa itu π , dan apa itu r .

Begitu juga dengan programming, kode-kode program jumlahnya sangat banyak. **HTML** baru satu bidang, belum lagi **CSS**, **PHP**, **JavaScript**, **MySQL**, dan kode program lanjutan lain seperti **Bootstrap**, **jQuery**, dll.

Jadi, silahkan nikmati “belajar coding”. Tidak usah khawatir harus menghapal setiap kode. Saya pun masih sering membuka buku ini jika lupa sesuatu, padahal ini semua saya yang tulis :)

Contoh Kode Program

Seluruh contoh kode program yang ada di buku **HTML Uncover** bisa di download dari folder sharing *Google Drive* yang saya kirim pada saat pembelian: **belajar_html.zip**.

Sebagaimana yang nantinya akan anda pelajari, halaman web lengkap diawali dengan kode HTML seperti `<!DOCTYPE html>` serta tag-tag HTML lain seperti `<head>`, `<title>` dan `<body>`. Namun agar menghemat tempat, dalam buku ini kode pembuka HTML tersebut tidak selalu saya tulis. Di dalam file *belajar_html.zip* saya kembali melengkapi tag-tag ini.

File kode program saya kelompokkan ke dalam folder sesuai dengan penomoran bab: `bab_4_struktur_dasar_html`, `bab_5_head_element`, `bab_6_text_element`, dst. Nama file juga akan disesuaikan berdasarkan nomor urut dan judul pembahasan.

Penomoran baris pada contoh kode program dalam buku ini (*line numbering*) berguna untuk memudahkan pembahasan. Jika anda ingin men-copy kode ini langsung dari eBook pdf, gunakan tombol **ALT + drag** agar nomor-nomor ini tidak ikut terseleksi.

Alternatif yang lebih saya sarankan adalah dengan mengetik ulang seluruh kode program. Tujuannya agar anda lebih cepat paham sekaligus bisa menghafal fungsi dari kode-kode tersebut.



Jika anda mencoba menulis ulang kode program yang ada di buku, dan ternyata tidak jalan, besar kemungkinan terdapat penulisan yang salah.

Di dalam programming, 1 karakter saja yang kurang (apakah itu berupa titik, tanda koma, atau tanda '>'), kode program tidak akan jalan sempurna. Jika ini yang terjadi, silahkan anda samakan dengan file-file dalam folder **belajar_html.zip** ini.

1. Berkenalan Dengan HTML

HTML adalah dasar dari seluruh halaman web yang ada di Internet. Dalam bab pembuka buku HTML Uncover ini saya akan mengajak anda untuk berkenalan dengan HTML. Kita akan mempelajari apa itu HTML, fungsi HTML, serta melihat sekilas cara penggunaan HTML dalam situs online.

1.1 Pengertian HTML

Walaupun saya yakin sebagian besar dari anda telah mengenal apa itu HTML, atau sudah pernah membuat beberapa kode HTML, tidak ada salahnya kita membahas sekilas tentang pengertian HTML. Barangkali butuh untuk tugas / bahan presentasi :)

HTML merupakan singkatan dari **Hypertext Markup Language**. Singkatan ini terdiri dari 3 komponen kata, yakni: **Hypertext**, **Markup** dan **Language**.

Kata **Hypertext** dari HTML menekankan pengertian: *text yang lebih dari sekedar teks* ('hyper'-text). Maksudnya selain berfungsi sebagai teks biasa, sebuah teks di dalam HTML juga bisa berfungsi sebagai penghubung ke halaman lain atau dikenal dengan istilah **link**. Nantinya kita juga akan melihat bahwa tidak hanya teks saja yang bisa digunakan sebagai link, tetapi bisa berupa gambar. Link inilah yang menjadi inti dari HTML.

Kata kedua dari singkatan HTML adalah **Markup**. Markup dapat diterjemahkan sebagai tanda atau penanda (bahasa inggris: *mark*). Di dalam HTML, kita akan menggunakan tanda-tanda khusus seperti `<p>`, `<a>`, atau ``. Tanda ini diperlukan untuk mengatur format dan membuat struktur halaman web.

Bagian terakhir dari HTML adalah **Language**. Istilah *language* jika diterjemahkan berarti: *bahasa*. Khusus bagi anda yang pernah berkenalan dengan bahasa pemrograman komputer, disini HTML tidak menggunakan '*Programming Language*', tetapi hanya '*Language*' saja.

Hal ini secara tidak langsung menyatakan bahwa HTML **bukanlah sebuah bahasa pemrograman**. HTML tidak memiliki struktur dasar seperti *variabel*, *kondisi IF*, *function*, atau *class* seperti layaknya sebuah bahasa pemrograman komputer.



Walaupun menyebut HTML sebagai sebuah *bahasa pemrograman* secara teknis tidak-lah 'pas', namun untuk menyederhanakan konsep belajar, istilah tersebut masih saya gunakan di dalam buku ini.

Merangkum penjelasan diatas, dapat disimpulkan bahwa HTML adalah sebuah bahasa khusus yang ditulis menggunakan tanda-tanda (**mark**) untuk membuat halaman web.

Agar lebih lengkap, mari kita lihat definisi HTML dari [wikipedia](#)¹:

“HTML or HyperText Markup Language is the standard markup language used to create web pages.”

Terjemahan bebasnya:

“HTML atau HyperText Markup Language adalah bahasa markup standar yang digunakan untuk membuat halaman web.”

Ternyata lebih sederhana dari penjelasan saya diatas :)

1.2 Fungsi HTML

Seperti yang kita ketahui bahwa HTML digunakan untuk membuat halaman web. Tetapi apa sebenarnya fungsi dari HTML?

Dalam proses *web development* (proses pembuatan web), HTML berfungsi untuk membuat struktur dari sebuah website. HTML digunakan untuk menandai bagian mana yang akan menjadi judul artikel, bagian mana yang berfungsi sebagai isi artikel, atau bagian mana yang butuh disajikan dalam bentuk tabel.

Mengenai tampilan dari website tersebut (misalnya apakah akan menggunakan warna *background* biru atau merah), diserahkan kepada teknologi web lain yang dikenal dengan CSS (*Cascading Style Sheet*).

Saat ini, untuk membuat sebuah *web modern*, tidak bisa hanya menggunakan HTML + CSS saja, tetapi juga memerlukan bantuan *bahasa pemrograman* web lain seperti JavaScript. Di sini saya menggunakan istilah ‘*web modern*’, karena kita memang bisa membuat website dari HTML saja (tanpa CSS dan JavaScript), tetapi web yang dihasilkan akan terkesan ‘jadul’ karena tidak memiliki tampilan yang cantik (sesuai standar saat ini).



Sama seperti HTML, CSS sebenarnya juga bukan bahasa pemrograman, tetapi adalah ‘bahasa kode’ yang mirip dengan HTML.

Beberapa pertanyaan yang sering muncul ketika baru pertama kali mempelajari HTML adalah bagaimana cara menampilkan bingkai (*border*) dari gambar atau bagaimana cara membuat *menu bar* yang muncul sendiri ketika mouse berada diatasnya (*menu rollover*)?

Semua efek tampilan dan interaksi seperti ini tidak bisa dilakukan dengan HTML saja (terutama untuk *menu rollover*), tetapi dibuat dengan gabungan HTML + CSS + JavaScript.

Dilihat dari sisi fungsinya, ketiga teknologi web ini berbagi peran masing-masing. HTML digunakan untuk membuat struktur konten atau isi dari halaman web. CSS berfungsi untuk memperindah tampilan, sedangkan JavaScript berperan untuk membuat interaksi.

¹<http://en.wikipedia.org/wiki/HTML>

Sebutan kerennya: **HTML for content, CSS for presentation and JavaScript for behavior.** Ketiga teknologi inilah yang mendasari semua website yang ada di internet saat ini.

Walaupun HTML sepertinya tidak terlalu *powerful*, tetapi ia adalah dasar dari semuanya. Baik CSS maupun JavaScript harus memiliki struktur HTML untuk dapat bekerja. Bahasa pemrograman web lain seperti **PHP** atau **ASP** juga digunakan untuk menghasilkan HTML. Sehingga HTML memiliki peran paling penting dan mutlak harus dikuasai sebelum masuk ke dalam teknologi web lain seperti **CSS**, **JavaScript**, atau **PHP**.



Dalam buku ini saya akan fokus membahas tentang HTML, namun seperti yang dijelaskan diatas, HTML sangat berkaitan dengan CSS dan JavaScript. Oleh karena itu, saya akan menggunakan CSS dan JavaScript ketika membahas beberapa fitur HTML. Sebagai bahan perkenalan dengan kedua teknologi ini, terdapat bab khusus di akhir buku yang membahas sekilas tentang cara penggunaan CSS dan JavaScript.

Secara bertahap saya akan melengkapi buku-buku web programming DuniaIlkom. Saat ini sudah tersedia buku [CSS Uncover^a](#), [PHP Uncover^b](#), [JavaScript Uncover^c](#) dan [MySQL Uncover^d](#). Semua buku ini akan menuntun anda untuk mempelajari materi dasar web programming.

Meskipun terlihat cukup banyak, kelima materi ini barulah materi dasar. Jika ingin serius menjadi seorang web programming professional (untuk dunia kerja), harus lanjut ke materi advanced seperti **Bootstrap**, **jQuery**, **Sass**, **Code Igniter**, **Laravel**, dll. Penjelasan lebih lengkap bisa ke sini: [Ingin Belajar Web Programming, Harus Mulai Dari Mana^e](#)?

^a<http://www.duniaIlkom.com/css-uncover-panduan-belajar-css-lengkap-untuk-pemula/>

^b<http://www.duniaIlkom.com/php-uncover-panduan-belajar-php-lengkap-untuk-pemula/>

^c<http://www.duniaIlkom.com/javascript-uncover-panduan-belajar-javascript-untuk-pemula/>

^d<http://www.duniaIlkom.com/mysql-uncover-panduan-belajar-mysql-dan-mariadb-untuk-pemula/>

^e<http://www.duniaIlkom.com/ingin-belajar-web-programming-harus-mulai-dari-mana/>

1.3 Melihat Sekilas HTML

Salah satu hal yang unik dari **HTML** (juga **CSS** dan **JavaScript**) adalah kita bisa melihat dengan bebas kode-kode yang digunakan untuk membuat sebuah halaman web.

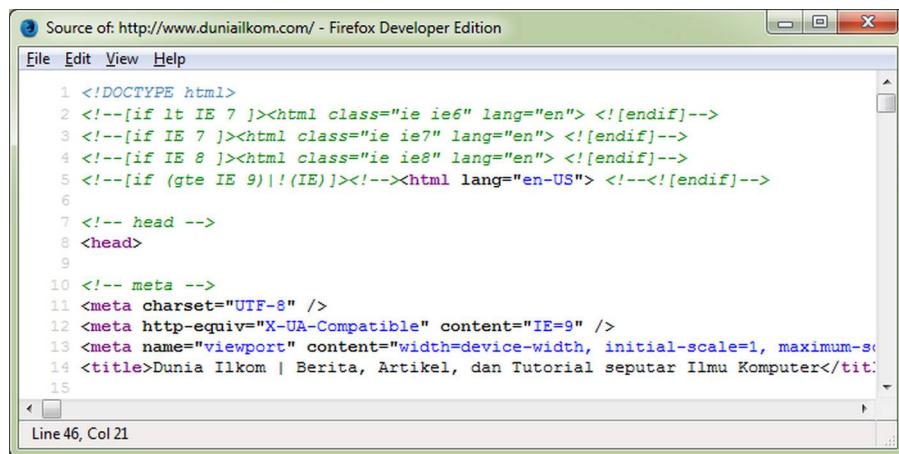
Caranya, silahkan jalankan web browser (*Google Chrome* atau *Mozilla Firefox*), buka sebuah situs web, kemudian klik kanan pada halaman situs dan pilih **View page source**. Sebagai contoh, di dalam gambar dibawah ini anda dapat melihat kode-kode yang digunakan untuk membuat halaman home dari situs [DuniaIlkom](#)².

²<http://www.duniaIlkom.com>



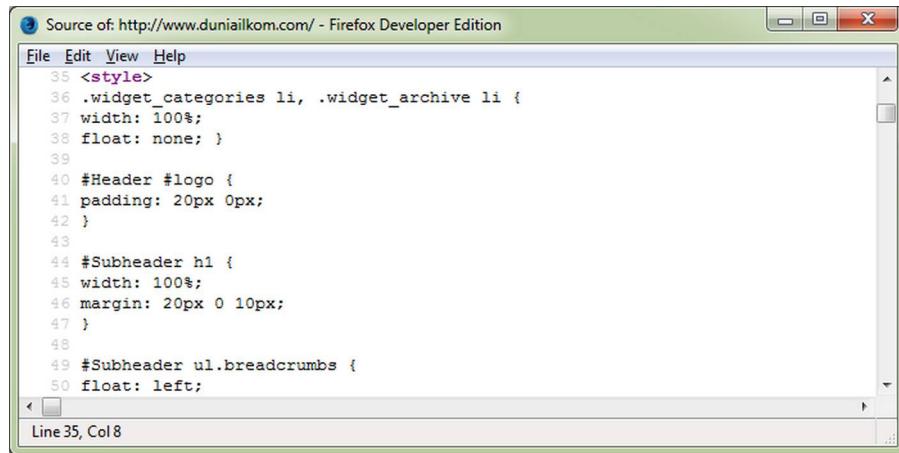
Gambar: Cara melihat source code website DuniaIlkom

Dalam tampilan ini anda dapat melihat (dan juga bisa mempelajari) seluruh kode HTML, CSS dan JavaScript yang digunakan untuk membuat situs duniaIlkom.



Gambar: Kode HTML pada situs DuniaIlkom

Bagian yang ditulis dengan tanda kurung siku seperti `<!DOCTYPE html>`, `<head>`, atau `<div>` merupakan kode-kode HTML. Lebih jauh lagi, anda juga dapat melihat kode CSS dan JavaScript yang semuanya berada di dalam 1 halaman.



Gambar: Kode CSS pada situs Duniailkom

Kombinasi dari HTML, CSS dan JavaScript inilah yang menghasilkan halaman *home* dari DuniaIlkom. Jika anda lanjutkan *scroll* sampai baris terakhir, anda akan melihat angka 750an pada sisi kiri bawah, yang menunjukkan bahwa untuk membuat halaman ini, membutuhkan 750an baris kode program!



The screenshot shows the 'Source of: http://www.duniaIlkom.com/' window in Firefox Developer Edition. The code is a snippet of JavaScript, likely a plugin or theme script, with line numbers 667 to 682 visible. The code creates two link elements, 'corecss' and 'themecss', and inserts them into the head of the document. It handles both standard attribute assignment and the newer `setAttributे` method. The code is written in a mix of standard JavaScript and some specific syntax, possibly from a plugin like SyntaxHighlighter.

Gambar: Kode JavaScript pada situs DuniaIlkom

Namun, jangan takut. Walaupun terkesan ‘rumit’, HTML sangat sederhana sekaligus sangat *powerfull*. Membuat sebuah website memang memerlukan skill yang mumpuni dan menguasai berbagai jenis bahasa pemrograman. Selain HTML, seorang web programmer setidaknya juga harus paham tentang CSS, JavaScript, PHP dan MySQL.

Dalam buku ini kita akan membahas HTML secara detail yang diperlukan sebagai persiapan dan dasar bagi anda untuk mempelajari bahasa pemrograman web lainnya.

1.4 Mengenal Jenis-jenis Bahasa Pemrograman Web

Walaupun buku ini hanya akan membahas tentang HTML, tetapi saya juga ingin memperkenalkan anda dengan teknologi-teknologi pemrograman yang dibutuhkan untuk membuat website.

Saya rasa informasi ini cukup penting terutama jika anda baru pertama kali mengenal HTML dan ingin fokus menjadi **web programmer**.

Seperti yang anda lihat dari *source code* DuniaIlkom diatas, menguasai HTML saja tidaklah cukup. Dengan jujur saya katakan bahwa jika anda telah selesai membaca buku ini, itu barulah awal dari berbagai teknologi pemrograman web yang harus dikuasai.

Selain HTML, bahasa pemrograman lain yang ‘wajib dipahami’ adalah CSS, JavaScript, PHP dan database MySQL. Kelima teknologi ini menjadi dasar untuk dapat membuat website modern.

Sebagai contoh, di dalam buku ini nantinya saya akan membahas tentang cara membuat **form** dengan HTML. Form biasanya digunakan untuk proses registrasi atau pendaftaran di dalam sebuah situs.

Walaupun struktur form dibuat menggunakan HTML, tetapi untuk memperindah tampilan form kita harus menggunakan CSS. JavaScript berguna untuk menampilkan pesan error secara interaktif ketika user salah memasukkan beberapa data di dalam form.

Ketika form selesai di input, untuk dapat memproses isi dari form tersebut, kita harus menggunakan bahasa pemrograman **PHP**. Sebagai langkah akhir, tentu saja kita ingin menyimpan isi form tersebut ke dalam database **MySQL** untuk dapat diakses kembali di lain waktu.

Dari alur diatas, dapat dilihat bahwa baik **HTML**, **CSS**, **JavaScript**, **PHP** dan **MySQL** saling berhubungan. Kelimanya teknologi web ini secara bersama-sama digunakan untuk menghasilkan web yang modern dan interaktif.

Sama seperti disiplin ilmu yang lain, pengetahuan dasar yang kuat akan membantu anda untuk menguasai berbagai aspek seputar cara membuat website, dan sebagai langkah pertamanya adalah dengan mempelajari **HTML**.

HTML sendiri adalah teknologi yang telah ada selama puluhan tahun dan terus dikembangkan. Saat ini versi HTML terakhir adalah **HTML5**. Agar lebih mengenal tentang HTML, dalam bab selanjutnya saya akan menjelaskan tentang **sejarah dan perkembangan versi HTML**.

2. Sejarah dan Perkembangan HTML

HTML ikut menentukan arah dari Internet. Berbagai teknologi silih berganti hingga akhirnya menjadi HTML5 yang dikenal sekarang.

Dalam bab ini saya mencoba membawa anda untuk mengenal sejarah HTML, mulai dari awal kemunculannya, serta berbagai aspek teknologi yang menyertai perkembangan HTML. Selain itu juga akan dibahas tentang perkembangan web browser dan era '*'browser war'* yang cukup fenomenal.

Sejarah HTML ini saya tulis dengan cukup panjang dan detail. Jika anda sudah tidak sabar ingin menulis HTML, silahkan lanjut ke bab berikutnya. Akan tetapi dari cerita ini kita bisa mengenal istilah-istilah teknologi yang menyertai perkembangan HTML, seperti SGML, XML, XHTML, W3C dan WHATWG.

2.1 Sejarah HTML

Tim Berners-Lee dan CERN

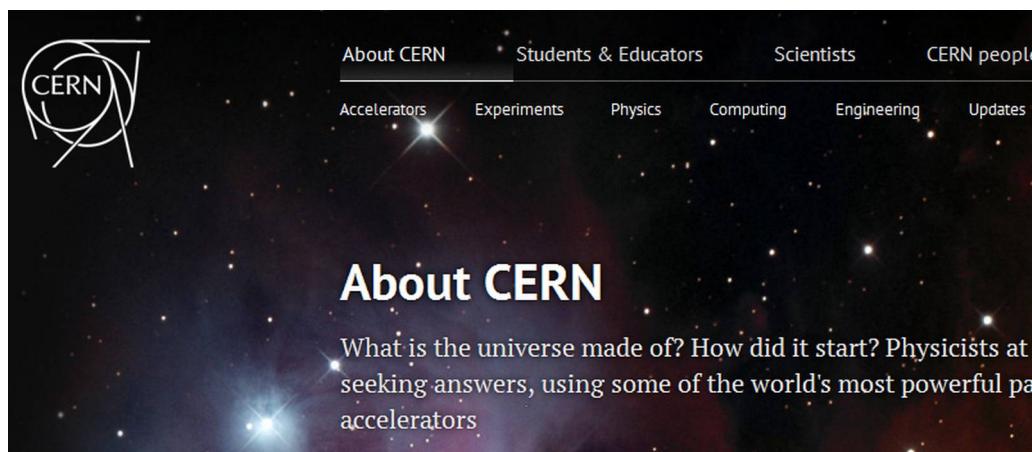
HTML dirancang pertama kali pada tahun 1989 oleh **Tim Berners-Lee**. Beliau merupakan seorang ilmuwan inggris yang saat itu bekerja di **CERN**. CERN sendiri sebenarnya bukanlah sebuah perusahaan teknologi atau organisasi yang berkaitan dengan internet.

CERN adalah singkatan dari bahasa prancis: *Conseil Européen pour la Recherche Nucléaire*, atau terjemahan bebasnya: *Komisi Eropa untuk Penelitian Fisika Nuklir*. CERN terletak di perbatasan Swiss dan Perancis, yang dibentuk pada tahun 1954 sebagai lembaga penelitian bersama negara Eropa untuk Fisika Nuklir.



Jika anda peminat iptek, CERN adalah tempat dimana *particle accelerator* terbesar di dunia berada, atau yang dikenal dengan **Large Hadron Collider (LHC)**. Disinilah tempat penelitian fisika quantum dilakukan oleh ribuan peneliti dari seluruh dunia. Pada tahun 2012 yang lalu, CERN menjadi pusat perhatian kalangan akademis karena berhasil menemukan partikel *higgs boson* atau *god particle* yang selama ini menjadi teka-teki di dunia fisika.

Ide tentang HTML dikembangkan oleh Tim Berners-Lee dengan tujuan memudahkan sesama ilmuwan CERN untuk berbagi hasil penelitian. Peneliti CERN melibatkan berbagai universitas di banyak negara, sehingga perlu sebuah cara agar hasil penelitian dari sebuah universitas bisa dilihat dengan mudah oleh universitas lainnya.



Gambar: Tampilan situs CERN - home.web.cern.ch

Awal lahirnya HTML

Pada akhir tahun 1991, Tim Berners-Lee mempublikasikan dokumen yang berjudul: **HTML Tags**. Dalam dokumen ini, Tim membuat konsep sederhana tentang HTML yang terdiri dari 18 *element*. Daripada membuat standar bahasa baru, Tim Berners-Lee merancang HTML yang didasarkan kepada konsep *bahasa markup* yang dikenal dengan **SGML** (*Standard Generalized Markup Language*).

Penulisan tag HTML dengan kurung siku berasal dari konsep SGML ini. Dengan kata lain, HTML adalah salah satu implementasi dari SGML. SGML merupakan sebuah standar internasional untuk membuat dokumen dengan tanda (*mark*) seperti *paragraf*, *list*, *heading*, dan lain-lain. Tag HTML seperti `<title>`, `<p>`, ``, dan `<h1>` sampai `<h6>` berasal dari SGML.

Namun tidak seluruh fitur yang ada di dalam HTML berasal dari SGML, salah satunya adalah: **hypertext link**. Atribut *href* di dalam tag `<a>` adalah murni hasil pemikiran Tim Berners-Lee. Inilah yang membedakan HTML dengan SGML.

Untuk menjalankan HTML, Tim Berners-Lee mengembangkan web browser pertama yang dinamakan dengan **WorldWideWeb** yang kemudian berganti nama menjadi **Nexus**. Ide tentang HTML ini dipublikasikan di dalam sebuah *mailing list* dan segera menjadi perhatian berbagai ilmuwan komputer di seluruh dunia.



Gambar: Tim Berners-Lee (2014) - sumber: wikipedia

HTML 1.0

Pada tahun 1993, web browser lain yang digunakan untuk menampilkan HTML mulai bermunculan. Beberapa diantaranya adalah: **Lynx**, **Mosaic** dan **Arena**. Tetapi karena belum tersedia sebuah standar baku bagaimana aturan HTML ditulis, masing-masing web browser mendefinisikan versi HTML mereka sendiri.

Untuk mengatasi hal ini, Tim Berners-Lee dan rekannya **Dave Raggett**, mengajukan sebuah proposal berjudul: **Hypertext Markup Language, Ver 1.0** kepada badan standarisasi **IEFT** (*Internet Engineering Task Force*). IEFT adalah badan standar internasional yang menangani arsitektur internet. Proposal inilah yang menjadi **HTML versi 1.0**.

Namun draft HTML 1.0 ini gagal menjadi standar karena melewati batas waktu yang ditentukan oleh IEFT. Oleh karena itu, HTML versi 1.0 tidak pernah menjadi sebuah standar resmi.

HTML 2.0

Karena draft HTML 1.0 mengalami permasalahan, pada awal 1994 IEFT membentuk **HTML Working Group** (disingkat menjadi **HTMLWG**). HTMLWG bertujuan untuk menyempurnakan HTML yang sebelumnya diusulkan oleh Tim Berners-Lee dan Dave Raggett.

Pada Juli 1994, HTML 2.0 resmi menjadi standar HTML pertama yang di-sah-kan oleh IEFT. HTML 2.0 memiliki spesifikasi yang didasari kepada *draft* HTML 1.0 dengan penambahan beberapa fitur baru yang telah banyak digunakan pada web browser saat itu.

Sebagai contoh, web browser **Mosaic** yang menjadi web browser paling banyak digunakan, menambahkan tag `` untuk menampilkan gambar. Tag `` ini kemudian menjadi bagian dari standar HTML 2.0.

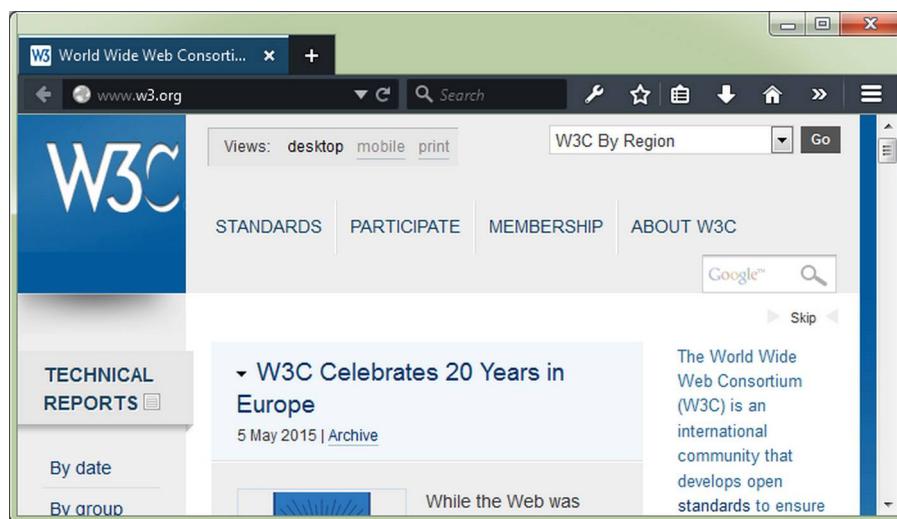


Web browser **Mosaic** dikembangkan oleh NCSA (*National Center for Supercomputer Applications*), sebuah lembaga penelitian dari University of Illinois, USA.

Pada tahun yang sama, Tim Berners-Lee membentuk **World Wide Web Consortium (W3C)**, dengan misi mempopulerkan penggunaan HTML. **W3C**¹ dan HTMLWG merupakan 2 organisasi

¹<http://www.w3.org/>

yang terpisah dan sama-sama menangani standar HTML. Kedua badan ini pada awalnya mencoba untuk saling bekerja sama.



Gambar: Tampilan situs W3C - www.w3c.org

Perang Browser 1 (Browser War)

Pada November 1994, **Marc Andreessen**, salah satu programmer web browser **Mosaic** keluar dari NCSA dan mendirikan perusahaan **Mosaic Communications** yang kemudian berganti nama menjadi **Netscape Communications Corp**.

Netscape Communications Corp mengeluarkan web browser **Netscape** yang menjelma menjadi pemimpin pasar web browser pada saat itu. Tidak seperti sekarang di mana aplikasi web browser bisa di dapat dengan gratis, aplikasi web browser seperti Netscape merupakan aplikasi berbayar, sehingga banyak produsen software bersaing untuk menjadi yang terbaik.

Sebuah era yang dinamakan **browser war** (perang browser) dimulai pada pertengahan tahun 1995 saat Microsoft merilis web browser IE 1.0 (**Internet Explorer 1.0**).

Netscape dan IE memulai '*perang*' untuk merebut pasar web browser. Masing-masing pihak merilis berbagai fitur untuk membuat web browser mereka semenarik mungkin. Netscape merilis **JavaScript** sebagai fitur standar yang kemudian disaingi oleh **JScript** dari Internet Explorer. Masing-masing web browser juga mengimplementasikan tag-tag HTML non-standar, seperti `<blink>` dan `<marquee>`.

Era '*browser war*' ini menyulitkan proses pembuatan web dengan HTML, karena programmer harus membuat 2 buah kode program untuk masing-masing web browser. Ini diperlukan karena sebuah tag HTML yang berjalan di Netscape, bisa jadi tidak tersedia di IE, dan begitu juga sebaliknya. Standar HTML versi 2.0 perlu perubahan!

HTML 3.0

Melihat kebutuhan akan standar HTML yang baru, pada pertengahan tahun 1995, **Dave Raggett** yang saat itu bekerja sebagai peneliti di perusahaan komputer **Hewlett-Packard (HP)**, mencoba mengajukan draft HTML 3.0 kepada badan standarisasi IETF.

Akan tetapi karena draft HTML 3.0 mengalami banyak perdebatan (tentang fitur apa yang harus ada, dan fitur mana yang ingin dihapus, dll) HTML 3.0 akhirnya gagal menjadi standar.

Permasalahan ini kembali ditangani oleh HTMLWG yang mencoba merumuskan ulang HTML 3.0 (sama seperti pada kasus pada HTML 1.0). Akan tetapi HTMLWG juga mengalami jalan buntu, dan berujung dengan pembubaran badan ini di akhir tahun 1996.

HTML 3.2

Nasib perkembangan HTML kemudian ditentukan oleh **World Wide Web Consortium (W3C)** yang didirikan oleh Tim Berners-Lee. W3C mencoba memperbaiki draft HTML 3.0 dan akhirnya pada Januari 1997, standar **HTML 3.2** resmi dirilis oleh W3C (bukan lewat IETF sebagaimana standar HTML 2.0 sebelumnya).

HTML 3.2 menambahkan fitur baru seperti `<table>`. Dan telah mendukung pemisahan antara konten dengan tampilan. Untuk membuat desain tampilan website, W3C ‘*mengeluarkannya*’ dari HTML dan fungsi tersebut diserahkan kepada teknologi web lain yang dikenal dengan CSS.

Pada tahun 1997 ini juga, peta ‘*browser war*’ berubah secara dramatis. Pada Oktober 1997, Microsoft merilis Internet Explorer 4.0 yang dipaketkan ke dalam seluruh sistem operasi **Windows** secara gratis. Hal ini menandai dimulainya monopoli Microsoft di dalam pasar web browser. Sehingga secara perlahaan tapi pasti, Netscape mengalami kemunduran.

HTML 4.0 dan HTML 4.01

Pesatnya perkembangan internet dan HTML, membuat W3C langsung mengajukan draft HTML versi 4.0 pada akhir tahun 1997.

Pada Desember 1998, standar **HTML 4.0** resmi di rilis oleh W3C. HTML 4.0 menyempurnakan penulisan tag , dukungan penuh untuk CSS, dan menambahkan fitur multimedia. HTML 4.0 juga datang dengan 3 variasi: **Strict, Transitional, dan Frameset**.

Setahun kemudian, pada Desember 1999, **HTML 4.01** dirilis untuk menyempurnakan beberapa error yang terdapat pada HTML 4.0.

Perang web browser Netscape vs IE terus berkembang, keduanya berlomba menambahkan fitur baru selain standar HTML 4.01. Penulisan tag yang tidak standar juga banyak dipakai oleh kedua web browser.

Melihat hal ini, W3C kembali mengembangkan versi perbaikan dari HTML 4.01. Akan tetapi karena keterbatasan yang ada di dalam HTML, W3C mencoba beralih ke konsep bahasa markup lain, yang bernama **XML**. Tahun 1999 ini menjadi tahun dimana HTML mengalami ‘mati suri’, karena perkembangan HTML selanjutnya beralih ke **XHTML**.

XHTML 1.0

XHTML adalah singkatan dari (**eXtensible HyperText Markup Language**). Jika anda mengingat bahwa HTML menggunakan SGML sebagai dasar penulisan, maka XHTML mencoba keluar

dari SGML untuk mengadopsi bahasa markup yang lebih ‘ketat aturan’, yakni **XML (eXtensible Markup Language)**. Dengan kata lain, XHTML adalah versi HTML dari XML.

Baik SGML dan XML sama-sama berasal dari kelompok bahasa markup yang penulisannya menggunakan tag-tag dengan kurung siku, namun XML memiliki aturan yang lebih ketat. Di dalam XML, seluruh tag harus ditulis dalam huruf kecil, setiap tag harus di tutup, dan setiap nilai atribut harus ditulis dengan tanda kutip.

W3C berpendapat bahwa XHTML adalah bahasa web masa depan dan sebagai penerus dari HTML. W3C secara resmi mempublikasikan standar **XHTML 1.0** pada Januari 2000.

Tidak ada penambahan tag maupun fitur baru di dalam XHTML 1.0. Standar XHTML 1.0 hanyalah versi XML dari HTML 4.01. Seluruh tag yang ada pada HTML 4.01 juga tersedia di dalam XHTML 1.0.

XHTML 1.1

Segara setelah XHTML 1.0 dipublikasikan, W3C merilis standar **XHTML 1.1** pada Mei 2001. Versi XHTML 1.1 ini memperkenalkan modul-modul untuk memisahkan bagian halaman XHTML. Aturan penulisan dan tag-tag yang di dukung juga lebih ketat dari XHTML 1.0. Tag-tag dan atribut yang berfungsi untuk tampilan tidak lagi didukung, karena hal ini seharusnya dibuat dengan CSS.

Pada saat ini, gabungan XHTML dan CSS adalah aturan penulisan HTML yang ‘terbaik’. Berbagai buku dan referensi merekomendasikan penggunaan XHTML daripada HTML, karena dianggap lebih rapi dan ‘standar’.

Hasil Akhir Perang Browser 1

Perang browser juga sudah mereda dengan kekalahan telak Netscape. Pada tahun 2002, hampir 96% pasar web browser dimiliki oleh Microsoft dengan Internet Explorer. Era ini menandai akhir perang browser pertama yang dimenangi oleh Microsoft.

Hal ini juga mengakibatkan berhentinya penambahan fitur baru pada web browser. Selama 5 tahun (2001 – 2006), Microsoft hanya merilis 1 versi IE, yakni IE versi 6 yang menjadi web browser default pada **Windows XP**.

Setelah dikalahkan oleh Microsoft, Netscape kemudian merilis kode program web browser mereka dan memberikannya kepada sebuah badan non-profit: **Mozilla Foundation**.

XHTML 2.0

Melanjutkan perkembangan standar XHTML, selama Augustus 2002 sampai dengan Juli 2006, W3C mencoba membuat draft untuk generasi XHTML berikutnya, yakni **XHTML 2.0**.

XHTML 2.0 adalah versi yang sepenuhnya baru, dan mencoba memutus rantai dengan HTML. XHTML 2.0 tidak lagi mendukung fitur-fitur yang sebelumnya ada pada HTML.

Hal ini membuat mayoritas programmer web mengajukan protes. Menurut mereka, XHTML 2.0 tidak melihat kebutuhan web developer dan hanya berfokus kepada standar yang akan

susah diimplementasikan. Masalah ini membuat pembahasan mengenai XHTML 2.0 mengalami perdebatan panjang dan terhenti.

Di sisi web browser, setelah beberapa tahun Internet Explorer tidak memiliki kompetitor, penerus Netscape, **Mozilla Firefox 1.0** dirilis pada tahun 2004 dengan gratis. Firefox secara perlahan mulai menjadi web browser alternatif dari IE. Situs sosial media seperti *Digg*, *Facebook*, dan *Twitter* juga mulai bermunculan.

WHATWG dan HTML5

Melihat arah XHTML yang tidak jelas dan ‘berhenti’ pada draft XHTML 2.0, beberapa programmer dari *Apple*, *Mozilla Foundation* dan *Opera Software* mendirikan **Web Hypertext Application Technology Working Group**, yang disingkat menjadi **WHATWG** pada tahun 2004.

WHATWG berkeinginan untuk membuat standar HTML yang didasarkan kepada kebutuhan programmer dan web browser. Alih-alih menggunakan konsep XML dan XHTML, WHATWG memutuskan ‘membangkitkan’ kembali HTML 4.01 yang telah lama ditinggalkan.

WHATWG kemudian merancang draft **Web Forms 2.0** dan **Web Apps 1.0** yang kemudian digabung menjadi **HTML5**. WHATWG sendiri bukanlah organisasi pesaing dari W3C, mereka sebenarnya berencana merampungkan proposal HTML5 untuk kemudian diserahkan kepada W3C.

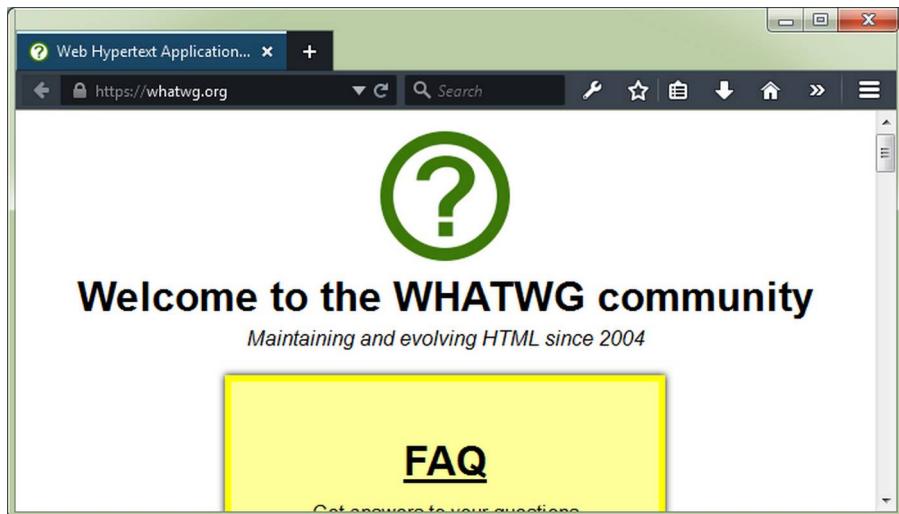
Sementara WHATWG mengembangkan HTML5, W3C tetap meneruskan XHTML 2.0. Namun pada Oktober 2006, Tim Berners-Lee mengakui perpindahan dari HTML ke XHTML tidak berjalan seperti yang diharapkan. Beberapa bulan kemudian, W3C membentuk tim khusus dalam mengembangkan HTML 5 dengan menggunakan dasar-dasar HTML5 dari WHATWG.

Sampai pada tahap ini, perkembangan HTML menjadi sedikit membingungkan, karena terdapat 3 versi kelanjutan HTML. Yang pertama dan kedua adalah **XHTML 2.0** dan **HTML 5** yang dikembangkan oleh tim W3C, dan yang ketiga adalah **HTML5** yang dikembangkan oleh WHATWG (perhatikan bahwa HTML5 dan HTML 5 adalah 2 versi yang berbeda, dan penulisan keduanya hanya dibedakan dengan sebuah spasi!)

Pada tahun 2009 W3C akhirnya memutuskan untuk menghentikan pengembangan XHTML 2.0 dan ikut mengembangkan HTML5 bersama WHATWG.

Perkembangan web browser juga menjadi sesuatu yang menarik. Saat itu Mozilla Firefox berbagi pangsa pasar dengan IE. Mozilla Firefox mulai populer karena memiliki fitur yang beragam daripada IE (terutama karena memiliki fitur *plugin*). Bahkan pada Desember 2009, **StatCounter** (sebuah situs survei teknologi internet) menyatakan bahwa Mozilla Firefox telah menjadi web browser paling populer di dunia, mengalahkan IE.

Pada 11 desember 2008, Google memutuskan untuk merilis web browser **Google Chrome**. Bersama-sama dengan IE dan Mozilla Firefox, era *browser war* kedua dimulai.



Gambar: Tampilan situs WHATWG - www.whatwg.org

HTML5 sebagai Living Standard

Terkait kelanjutan HTML5, WHATWG dan W3C saling bekerja sama untuk mengembangkan masa depan HTML. Pada Juli 2012, keduanya sepakat untuk ‘membagi tugas’. W3C mengembangkan standar tunggal HTML 5, sedangkan WHATWG mengerjakan apa yang disebut sebagai **Living Standard HTML5**.

Konsep *Living Standard* adalah standar ini tidak akan pernah selesai dan selalu diupdate dan disempurnakan. Dengan demikian fitur-fitur baru dapat langsung di terapkan tanpa menunggu pengesahan dari W3C.

Dari sisi web programmer, Living Standar HTML5 ini menjadi tantangan tersendiri. Karena mengharuskan kita untuk selalu update dengan fitur terbaru HTML5, serta mencari tahu web browser mana saja yang telah mendukung fitur tersebut.

Akhirnya setelah penantian yang panjang, pada tanggal 28 October 2014, standar HTML5 resmi dirilis oleh W3C.

HTML 5.1, HTML 5.2 dan HTML 5.3

Segera setelah standar HTML 5 diserahkan oleh WHATWG, W3C langsung bekerja untuk mengembangkan spesifikasi HTML 5.

Standar [HTML 5.1²](https://www.w3.org/TR/html51/) di rilis sebagai rekomendasi pada bulan November 2016 dan diupdate pada 3 October 2017. Isinya berupa perbaikan bug serta penambahan fitur seperti tag `<menu>` dan `<menuitems>`, input type *month*, *week* dan *datetime-local* untuk form, atribut `srcset` dan `sizes` untuk tag ``, `<picture>` element, serta `<dialog>` element.

Berikutnya pada 14 Desember 2017, W3C secara resmi merilis [HTML 5.2³](https://www.w3.org/TR/html52/). Selain perbaikan bug, tidak banyak penambahan tag baru dalam spesifikasi ini. Malah tag `<keygen>`, `<menu>`

²<https://www.w3.org/TR/html51/>

³<https://www.w3.org/TR/html52/>

dan <menuitems> yang sebelumnya ditambahkan oleh HTML 5.1 akan dihapus. Penambahan fitur baru lebih banyak ke dalam kategori advanced yang melibatkan JavaScript.

Pengerjaan standar [HTML 5.3⁴](#) segera dimulai setelah HTML 5.2. Rencananya, HTML 5.3 akan menjadi rekomendasi final pada akhir tahun 2018.

Fuihh... demikianlah sejarah panjang dan perkembangan HTML dari awal kemunculan hingga HTML5. HTML5 ibarat tokoh *hero* yang kembali bangkit setelah 15 tahun ‘*mati suri*’ karena ditinggalkan oleh W3C (sempat digantikan oleh XHTML).

2.2 Teknologi dan fitur baru dalam HTML5

Selain menyempurnakan versi sebelumnya, HTML5 juga membawa fitur baru ke dalam HTML. Jika selama ini HTML hanya digunakan untuk membuat struktur web, HTML5 membawa berbagai teknologi baru untuk membuat web menjadi lebih powerful. Kita akan melihat sekilas apa saja fitur-fitur baru yang dibawa oleh HTML5.

Secara garis besar, standar HTML5 terdiri dari 3 bagian: **HTML5 markup**, **HTML5 API**, dan **teknologi yang berkaitan dengan HTML5**.

1. **HTML5 markup** adalah konsep HTML yang telah ada selama ini, yakni kode yang digunakan untuk membuat struktur halaman web. Untuk fitur ini, HTML5 menambahkan berbagai tag baru seperti <header>, <footer>, <aside>, <figure>, <article>, <audio>, dan <video>.
2. **HTML5 API (Application Program Interface)** adalah modul teknologi yang relatif baru. Untuk dapat menggunakannya, kita harus menggunakan bahasa pemrograman JavaScript. Beberapa fitur HTML5 API adalah: *Geolocation*, *Drag/Drop*, *LocalStorage*, *Web Workers* dan *Server-Sent Events*.
3. Karena efek pemasaran HTML5 yang cukup ‘wah’, terdapat **beberapa teknologi yang sering dianggap menjadi bagian dari HTML5**. Teknologi tersebut seperti: CSS3, SVG, dan MathML. Walaupun sepenuhnya terpisah, teknologi ini digunakan bersamaan dengan HTML sehingga sering dianggap sebagai bagian dari HTML5.

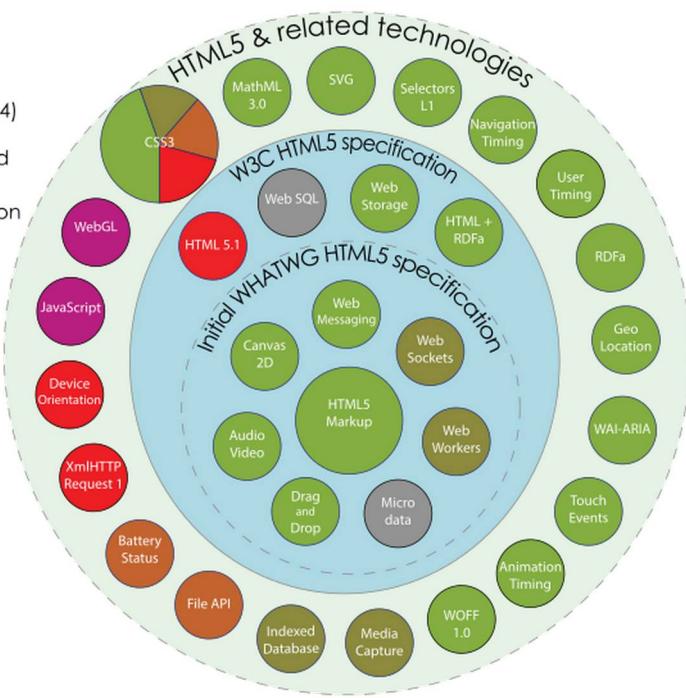
Berbagai teknologi yang melingkupi HTML5 ini dapat di lihat pada gambar 2.x berikut ini:

⁴<https://www.w3.org/TR/html53/>

HTML5

Taxonomy & Status (October 2014)

- Recommendation/Proposed
- Candidate Recommendation
- Last Call
- Working Draft
- Non-W3C Specifications
- Deprecated or inactive



Gambar: Fitur HTML5 dan teknologi yang menyertai perkembangan HTML5

2.3 Apa Yang Harus Saya Kuasai?

Seperti yang bisa kita lihat, HTML5 lebih dari sekedar bahasa markup untuk membuat struktur halaman web. HTML5 API dan berbagai teknologi lain yang sudah dijabarkan sebelumnya bisa membuat bingung dan sakit kepala.

Akan tetapi, jika anda adalah pemula yang ingin belajar pemrograman web, HTML yang perlu dipelajari adalah **HTML5 markup**, yakni bagian dari HTML5 yang berurusan dengan struktur halaman. Materi inilah yang akan kita bahas sepanjang buku nantinya.

Teknologi baru seperti **HTML5 API** merupakan fitur lanjutan yang perlu JavaScript. Materi seperti ini cukup susah bagi pemula. Selain itu mayoritas besar website juga tidak butuh HTML5 API. Fitur lanjutan ini baru bisa anda pelajari setelah punya basic JavaScript.

Dalam bab ini kita telah melihat sejarah panjang HTML dan melihat berbagai fitur baru yang dibawa oleh HTML5.

Berikutnya kita akan mempersiapkan ‘alat kerja’ (*tools*) untuk dapat mulai membuat web dengan HTML. Perangkat tersebut adalah **web browser** dan **text editor**.

3. Web Browser dan Text Editor

Sebelum mulai menulis kode HTML, kita harus mempersiapkan perangkat yang dibutuhkan, yakni **web browser** dan **text editor**. Terdapat beragam aplikasi web browser dan text editor yang bisa dipilih, dan sebagian besar diantaranya bisa didapat dengan gratis.

Dalam bab ini kita akan membahas tentang web browser dan text editor yang ‘layak’ digunakan untuk proses pembuatan web (*web development*).

3.1 Mempersiapkan Web Browser

Untuk menjalankan HTML, kita butuh web browser. **Web browser** adalah program yang akan menerjemahkan kode-kode HTML dan menampilkannya menjadi halaman website.

Saya yakin aplikasi web browser sudah terinstall di komputer anda, apakah itu **Google Chrome** atau **Mozilla Firefox**. Kedua web browser ini merupakan web browser terbaik untuk pengembangan web. Keduanya juga selalu mengikuti standar dan fitur terbaru HTML (selama anda rajin men-updatenya).

Selain Google Chrome dan Mozilla Firefox, masih terdapat web browser lain yang bisa digunakan, seperti **Internet Explorer**, **Opera** dan **Apple Safari**.

Walaupun tidak harus, saya menyarankan anda untuk menginstall kelima web browser ini. Tujuannya, untuk melihat bagaimana kode HTML yang kita buat ditampilkan pada masing-masing web browser.

Untuk mendownload web browser ini, silahkan mengunjungi situs resminya di alamat berikut: [Google Chrome¹](https://www.google.com/chrome/browser/), [Mozilla Firefox²](https://www.mozilla.org/en-US/firefox/new/), [Opera³](http://www.opera.com/download/), [Internet Explorer⁴](http://windows.microsoft.com/en-US/internet-explorer/download-ie), [Apple Safari⁵](http://support.apple.com/kb/DL1531).

Khusus untuk web browser *Apple Safari for Windows*, Apple telah lama tidak men-update web browsernya sehingga kemungkinan terdapat beberapa fitur yang belum didukung oleh web browser ini (Apple hanya mengupdate Safari untuk Mac OS dan iOS).

Web browser Internet Explorer (IE) juga sebaiknya tidak dipakai lagi dalam proses pengembangan web. Microsoft sendiri sudah meninggalkan IE dan menggantinya dengan [Microsoft Edge⁶](https://www.microsoft.com/en-us/windows/microsoft-edge) yang menjadi web browser default Windows 10. IE hanya cocok untuk proses percobaan, yakni melihat seperti apa tampilan halaman web untuk web browser “jadul”.

¹<https://www.google.com/chrome/browser/>

²<https://www.mozilla.org/en-US/firefox/new/>

³<http://www.opera.com/download/>

⁴<http://windows.microsoft.com/en-US/internet-explorer/download-ie>

⁵<http://support.apple.com/kb/DL1531>

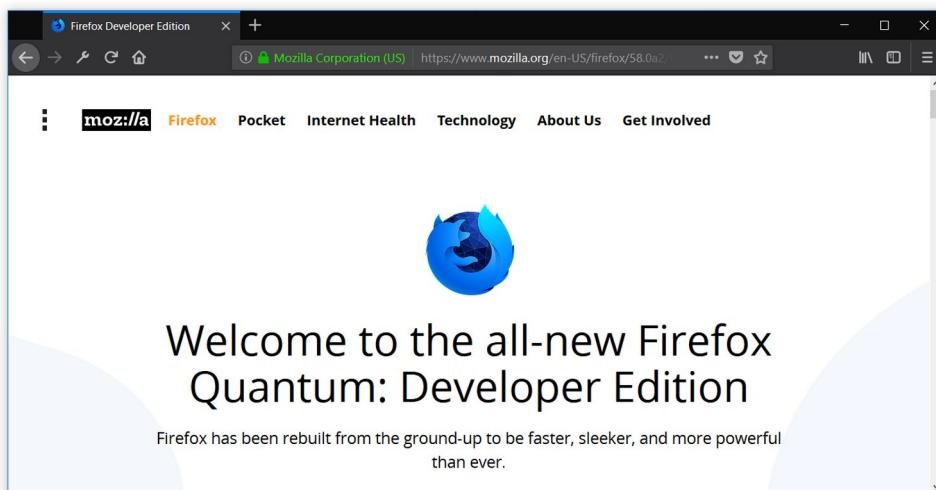
⁶<https://www.microsoft.com/en-us/windows/microsoft-edge>

Firefox Developer Edition

Mozilla juga merilis **Firefox Developer Edition**⁷ yang ditujukan khusus untuk web developer.

Secara umum tampilan dan fitur **Firefox Developer Edition** hampir sama dengan Firefox ‘biasa’ (plus dengan *theme* warna biru gelap). Mozilla mengatur shortcut yang berkaitan dengan pengaksesan kode agar lebih mudah dicapai.

Web browser ini sepenuhnya terpisah dari Firefox biasa, sehingga anda bisa memiliki 2 buah web browser Firefox secara bersamaan.



Gambar: Tampilan Firefox Developer Edition

Online installer vs Offline installer

Jika anda menginstall web browser dari link diatas, hampir semuanya berukuran kecil (dibawah 1 MB). File ini disebut juga dengan istilah **online installer**, yang artinya program kecil tersebut akan mendownload aplikasi sebenarnya pada saat proses instalasi. Umumnya aplikasi web browser berukuran sekitar 30-40 MB.

Apabila anda menginginkan file installer yang bisa diinstall tanpa koneksi internet, silahkan mencarinya di Google menggunakan keyword “**offline installer**” atau “**standalone installer**”. Sebagai contoh, untuk mencari installer Google Chrome, silahkan ketik di google: “*Google Chrome standalone installer*”.

Pangsa Pasar Web Browser

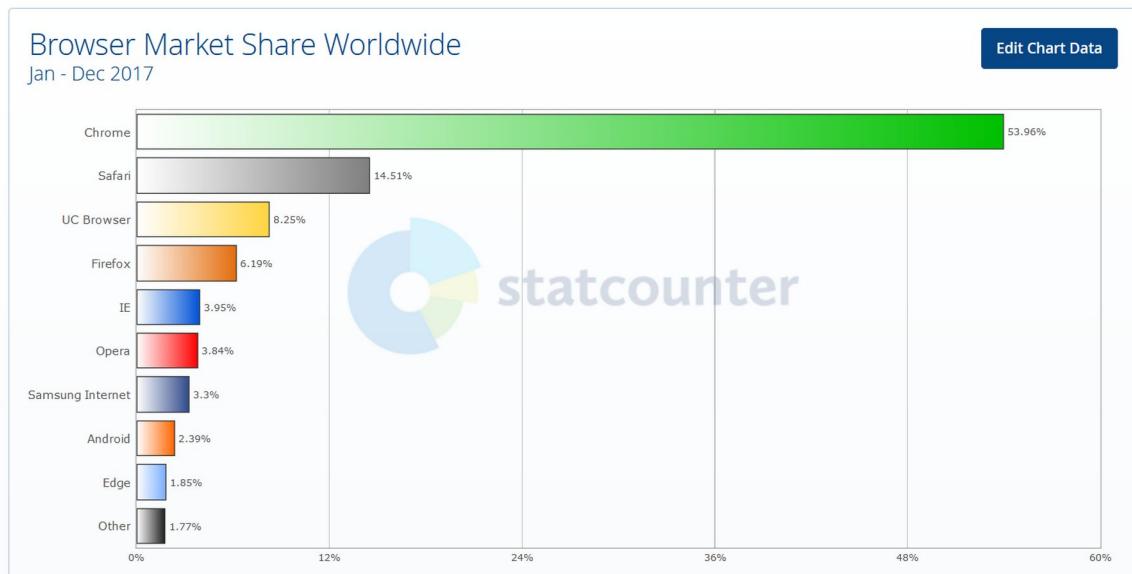
Web browser apa yang paling banyak digunakan saat ini? Jawaban singkatnya adalah: **Google Chrome**. Untuk mengetahui web browser apa saja yang paling banyak digunakan, kita bisa melihat hasil riset situs *trend web browser* seperti: [Statcounter](https://gs.statcounter.com/)⁸, [W3Counter](http://www.w3counter.com/globalstats.php)⁹ atau [Netmarketshare](http://www.netmarketshare.com/browser-market-share.aspx)¹⁰.

⁷ <https://www.mozilla.org/en-US/firefox/developer/>

⁸ <http://gs.statcounter.com/>

⁹ <http://www.w3counter.com/globalstats.php>

¹⁰ <http://www.netmarketshare.com/browser-market-share.aspx>

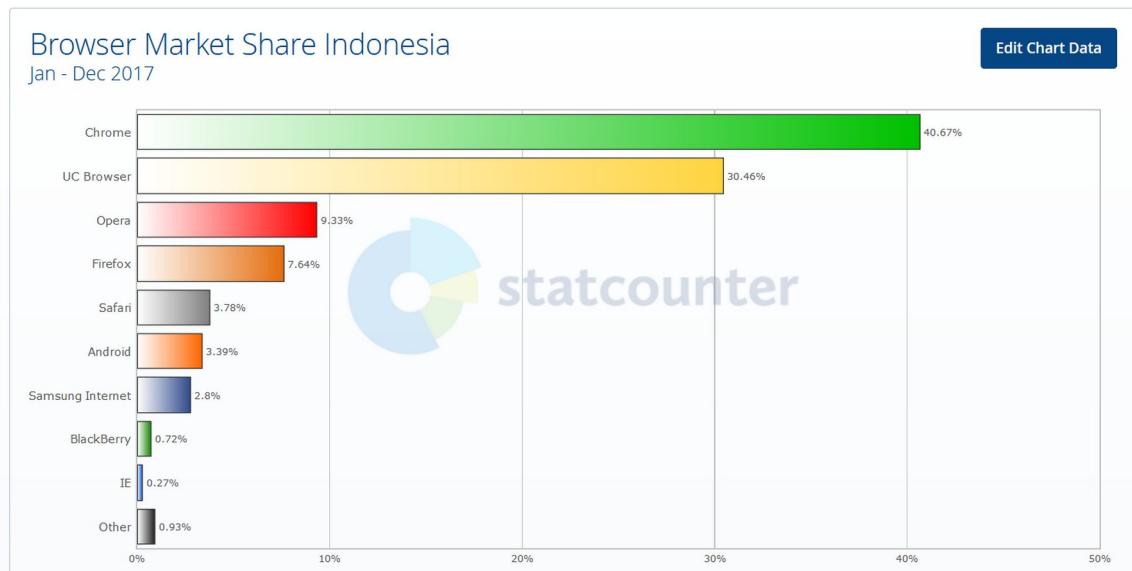


Grafik: Penggunaan web browser di dunia periode Januari 2017 - Desember 2017 (sumber: statcounter)

Grafik diatas adalah persentase penggunaan web browser sepanjang tahun 2017 (Januari - Desember) di seluruh dunia yang dianalisis oleh *Statcounter*. Hasil diatas sudah gabungan antara pengguna desktop dengan mobile.

Terlihat bahwa **Google Chrome** merupakan pemimpin pangsa pasar web browser saat ini dengan penggunaan 53% lebih. Artinya, setengah pengguna internet di dunia menggunakan Google Chrome. Setelah itu diikuti oleh **Safari** dan **UC Browser**. Besar kemungkinan kedua web browser ini berasal dari smartphone Apple serta smartphone android yang menggunakan UC Browser.

Bagaimana dengan Indonesia?

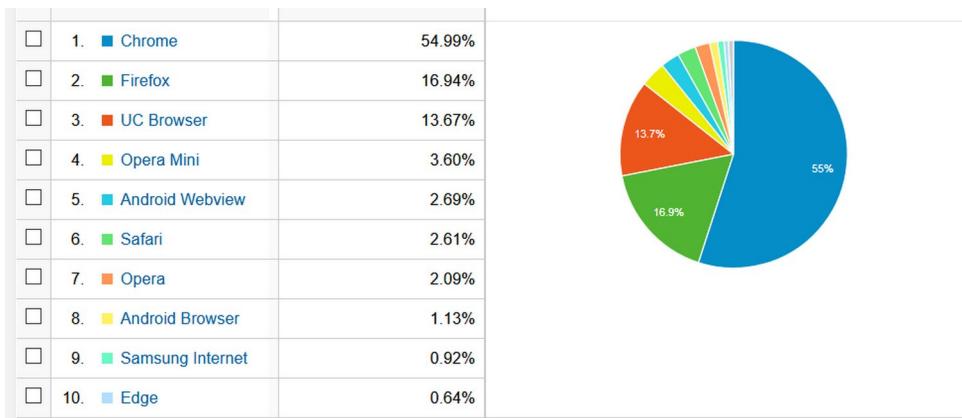


Grafik: Penggunaan web browser di Indonesia periode Januari 2017 - Desember 2017 (sumber: statcounter)

Hasilnya juga tidak berbeda. **Google Chrome** tetap menjadi web browser yang paling banyak

digunakan. Kemudian diikuti oleh **UC Browser** dan **Opera**.

Sebagai perbandingan, berikut data statistik web browser yang dipakai oleh pengunjung situs DuniaIlkom sepanjang tahun 2017:



Gambar: Statistik web browser pengunjung DuniaIlkom periode Januari 2017 - Desember 2017

Hasil ini seragam dengan riset dari Statcounter, dimana hampir setengah pengunjung duniaIlkom menggunakan **Google Chrome**. Hanya saja untuk posisi kedua dan ketiga dihuni oleh **Mozilla Firefox** dan **UC Browser**.

Dukungan Web browser untuk HTML5

Dalam buku ini, kita akan membahas banyak materi tentang HTML hingga fitur terbaru dalam HTML5. Khusus untuk ini, web browser ‘lawas’ tidak bisa memahami HTML5.

Jika anda menggunakan web browser seperti *Internet Explorer* 6 atau 7 untuk membuka halaman web yang dibuat dengan HTML5, terdapat kemungkinan halaman tersebut ditampilkan ‘*berbeda*’, bahkan gagal tampil sama sekali.

Sebagai pertimbangan, berikut adalah dukungan web browser secara umum untuk HTML5:

- Web browser **Google Chrome** dan **Mozilla Firefox** telah lama mendukung HTML5. Kedua web browser ini juga secara rutin mengeluarkan update dan memiliki fitur untuk update otomatis. Jika terdapat pengguna yang masih menggunakan versi lama, biasanya akan ditampilkan instruksi untuk segera mendownload versi terbaru. Secara garis besar, kedua web browser ini mendukung penuh HTML5.
- Walaupun tidak terlalu banyak digunakan, web browser **Apple Safari** dan **Opera** juga sudah lama mendukung HTML5. Versi ‘*tua*’ dari kedua web browser ini relatif jarang dijumpai, sehingga aman untuk HTML5.
- Web browser bawaan *smartphone* atau *tablet* mungkin memiliki keterbatasan dalam fitur HTML5. Namun karena perangkat ini terus berkembang dan oleh pembuat aplikasi (*google* dan *apple*) juga selalu mengikuti perkembangan. Secara perlahan web browser ini juga mendukung penuh HTML5.
- Web browser **Internet Explorer** (IE) mungkin akan menjadi masalah. IE versi 9 ke bawah tidak mendukung sebagian besar fitur HTML5. Masalah ini diperparah karena

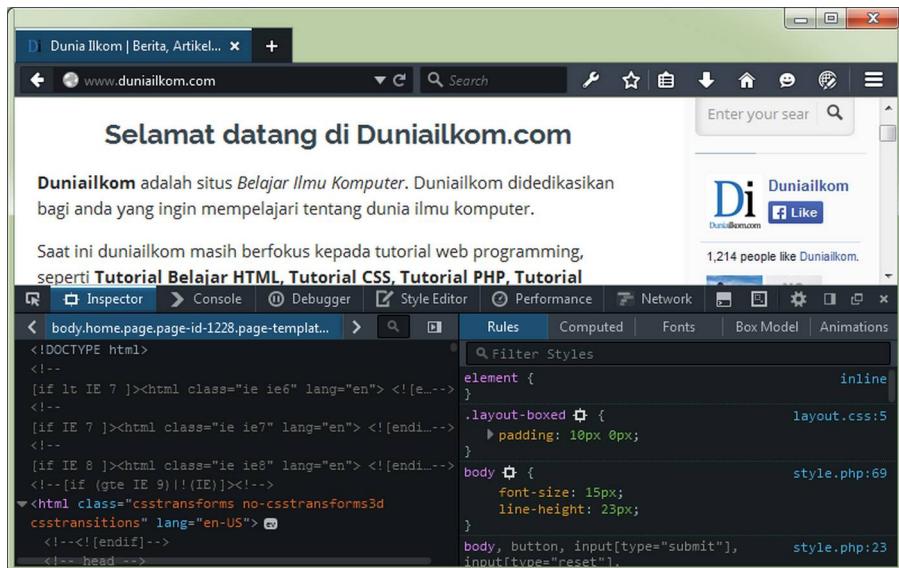
versi ‘*tua*’ dari IE masih banyak terinstall pada komputer Windows. Ditambah lagi pada beberapa sistem operasi Windows, IE tidak dapat di update ke versi terbaru. Sebagai contoh, *Windows XP* membawa IE versi 6, dan hanya bisa ditingkatkan hingga ke versi 8.

Mengenal Menu Developer Tools

Walaupun kita dapat melihat kode-kode HTML dari sebuah website dengan cara klik kanan dan memilih menu **view source**, web browser modern juga menyertakan menu khusus untuk menelusuri kode-kode HTML. Menu ini sering disebut dengan **Developer Tools**.

Cara paling cepat untuk mengaksesnya adalah dengan menekan kombinasi tombol **CTRL+Shift+i**. Silahkan anda melihat-lihat fitur yang ada. Fitur ini khusus dirancang untuk *web programmer* agar bisa melihat kode website dengan lebih mudah. Kita bahkan bisa mengedit langsung kode HTML dari menu developer tools.

Menu **Developer Tools** sangat berguna terutama jika anda sudah mempelajari CSS dan JavaScript. Karena fiturnya yang cukup banyak, anda perlu beberapa saat untuk mempelajarinya.



Gambar: Tampilan Developer Tools di Mozilla Firefox

3.2 Memilih Text Editor

Sebagaimana yang akan kita pelajari nanti, pada dasarnya file HTML hanyalah file teks biasa. Untuk membuat kode HTML, kita bisa menggunakan aplikasi **text editor**.

Text editor adalah aplikasi yang berfungsi untuk membuat dan mengedit file text. Salah satu contoh text editor adalah aplikasi **Notepad** bawaan Windows. Namun Notepad ini kurang cocok digunakan untuk pemrograman.

Terdapat banyak aplikasi text editor khusus pemrograman yang tersedia di internet dan mayoritas bisa didapat dengan gratis. Salah satunya adalah **Notepad++¹¹** yang akan saya gunakan

¹¹ <http://notepad-plus-plus.org>

sepanjang pembahasan buku ini. Saya memilih Notepad++ karena ringan dan berukuran kecil (hanya sekitar 5MB). Aplikasi ini sudah lebih dari cukup untuk membuat file HTML.

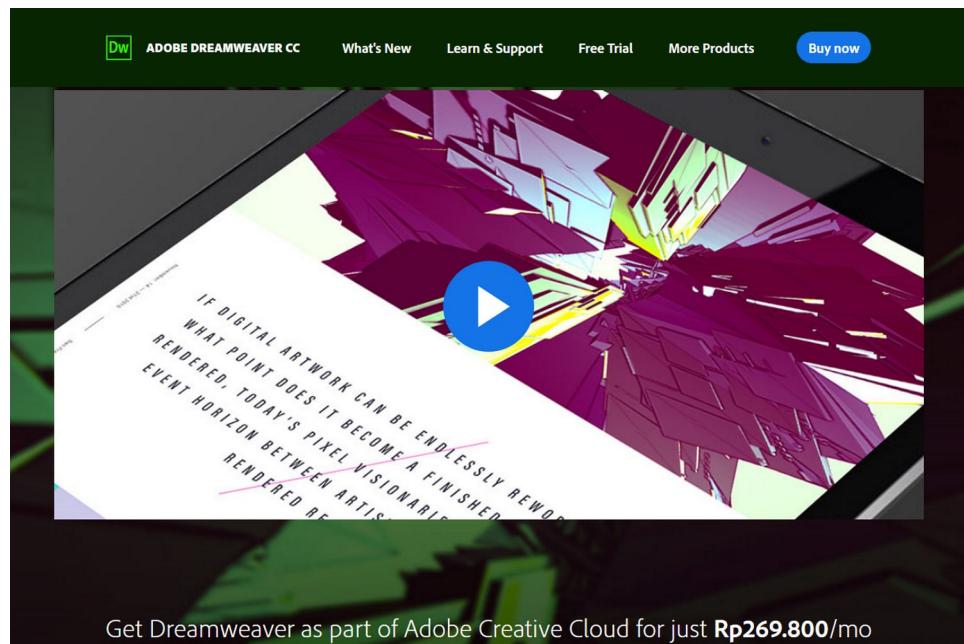
Sebagai alternatif, aplikasi text editor gratis lainnya adalah Atom¹², Komodo Edit¹³, Bracket¹⁴, VS Code¹⁵, atau Aptana Studio¹⁶. Untuk list lengkap mengenai aplikasi text editor dapat anda lihat di wikipedia.¹⁷

Adobe Dreamweaver

Salah satu aplikasi populer yang cukup ‘melegenda’ terutama bagi pemula web programming adalah **Adobe Dreamweaver**. Dreamweaver merupakan aplikasi canggih dengan fitur melimpah untuk pembuatan web.

Dreamweaver termasuk kelompok aplikasi yang dikenal dengan sebutan **WYSIWYG** (*What You See Is What You Get*), dimana kita bisa merancang tampilan website dengan cara ‘*drag and drop*’, yakni menggambar tampilan web secara visual tanpa harus mengetahui kode program dibalik itu (walaupun dreamweaver juga menyediakan fitur coding yang sangat lengkap).

Dibalik keunggulannya, menurut saya Dreamweaver tidak cocok untuk proses belajar. Aplikasi ini cenderung ‘berat’ dan berharga jutaan rupiah untuk versi legalnya. Dreamweaver lebih cocok digunakan jika anda telah memahami kode-kode program yang ada dibaliknya (dan mampu membeli versi original).



Gambar: Website resmi Adobe Dreamweaver CC

¹²<https://atom.io>

¹³<http://komodoide.com/komodo-edit>

¹⁴<http://brackets.io>

¹⁵<https://code.visualstudio.com>

¹⁶<http://www.aptana.com>

¹⁷http://en.wikipedia.org/wiki/List_of_text_editors

Saat ini versi terakhir dari Dreamweaver adalah **Adobe Dreamweaver CC¹⁸**. Aplikasi ini hanya bisa didapat dengan cara berlangganan dengan biaya Rp 269.800 per bulan.

Daripada menggunakan program bajakan, lebih baik mencari alternatif lain. Sangat jarang saya melihat lowongan kerja programmer yang mensyaratkan paham Dreamweaver. Bahkan akan menjadi nilai minus jika anda hanya bisa membuat program dari Dreamweaver saja.

Text Editor vs HTML Editor

Anda mungkin penasaran kenapa kita menggunakan **Text Editor**, bukannya **HTML Editor**. **HTML Editor** adalah aplikasi yang bisa membuat HTML dengan mudah bahkan tanpa perlu memahami kode-kode di dalamnya. Contoh HTML Editor adalah **Adobe Dreamweaver** atau **Microsoft Frontpage / Microsoft Expression Web** (tidak dikembangkan lagi).

Sama seperti Adobe Dreamweaver, aplikasi ini tidak ‘pas’ untuk belajar, karena akan menghasilkan kode-kode HTML secara instan. HTML editor lebih cocok jika anda telah mengetahui kode-kode HTML dan ingin dengan cepat membangun website.

Jika belum memahami HTML, aplikasi ini malah akan membuat bingung jika terjadi kesalahan atau jika ingin menambahkan fitur baru ke dalam halaman web.



Text Editor seperti Notepad++ tidak hanya bisa digunakan untuk HTML saja. Aplikasi ini juga bisa untuk membuat ‘hampir’ seluruh bahasa pemrograman lain, seperti PHP, JavaScript, CSS, dll.

3.3 Menjalankan File HTML Pertama Anda

Sebelum menutup bab ini, kita akan langsung mencoba menulis dan menjalankan file HTML. Teks editor yang saya gunakan adalah **Notepad++**. Jika anda belum menginstall aplikasi ini, silahkan download di <http://notepad-plus-plus.org/download>¹⁹.

Pada saat buku ini direvisi, versi terakhir dari Notepad++ adalah versi 7.5.4. Besar kemungkinan versi Notepad++ yang anda temui lebih baru dari ini.

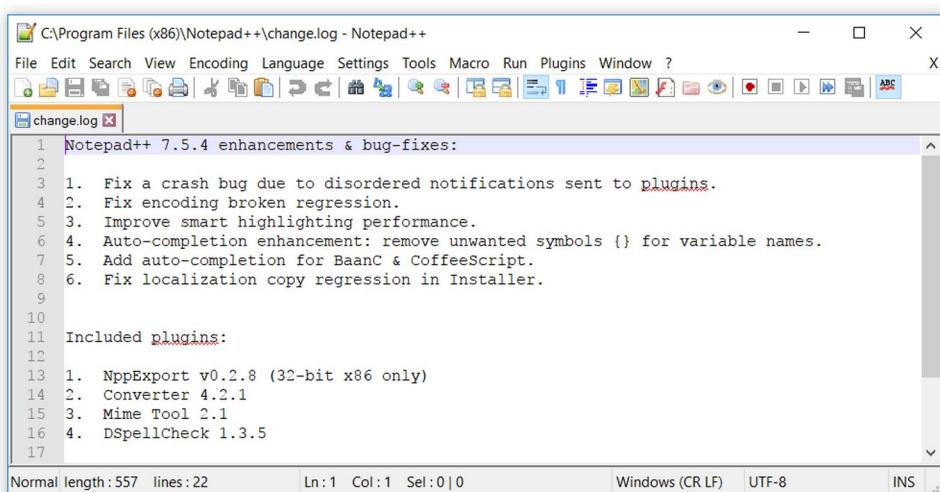
¹⁸<http://www.adobe.com/sea/products/dreamweaver.html>

¹⁹<http://notepad-plus-plus.org/download>



Gambar: Download aplikasi Notepad++

Proses instalasi Notepad++ hampir sama dengan program-program lain. Silahkan anda ikuti proses instalasi dan biarkan settingan default. Apabila sudah selesai, tampilan awal Notepad++ akan tampak seperti gambar dibawah ini.



Gambar: Tampilan awal Notepad++

Langkah berikutnya adalah mempersiapkan folder dimana file-file HTML akan diletakkan. Lokasi folder ini bebas dan tidak harus berada di dalam folder khusus. Agar mudah diakses, saya membuat folder **belajar_html** di drive D, sehingga alamatnya adalah: **D:\belajar_html**.

Selanjutnya, kembali ke aplikasi Notepad++, buat file baru, pilih menu **File->New** ketikkan kode berikut ini (atau boleh juga di copy-paste):

hello_world.html

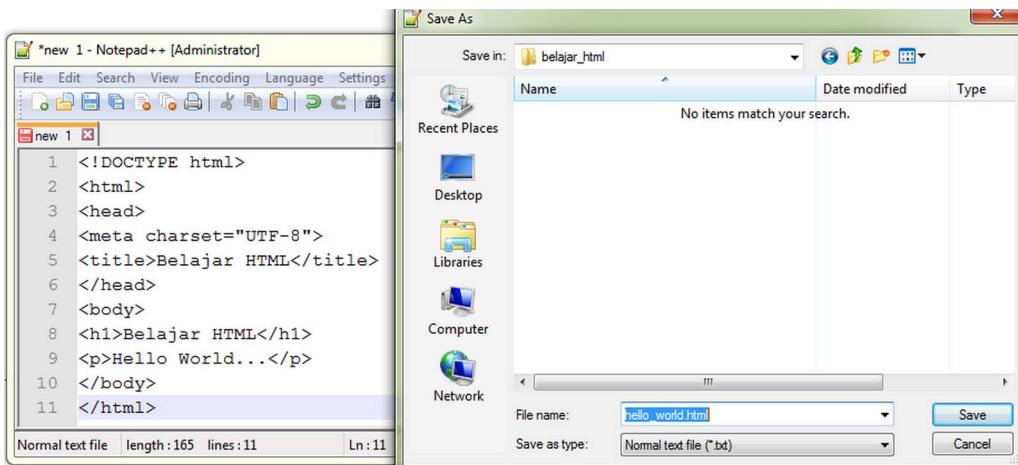
```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar HTML</title>
6 </head>
7 <body>
8   <h1>Belajar HTML</h1>
9   <p>Hello World...</p>
10 </body>
11 </html>
```



Jika anda memutuskan untuk men-copy paste kode program ke Notepad++, angka-angka di sisi kiri (line numbering) mungkin akan ikut tercopy. Jika menggunakan Adobe Acrobat Reader, silahkan tahan tombol ALT lalu seleksi kode diatas.

Pada tahap ini anda tidak perlu memahami kode program diatas, kita akan membahasnya dengan detail sepanjang buku ini. Langsung saja save dengan menekan tombol CTRL+S, atau pilih menu **File->Save**. Cari lokasi folder **belajar_html**, kemudian ganti nama file menjadi **hello_world.html** dan klik **Save**.



Gambar: Save file hello_world.html

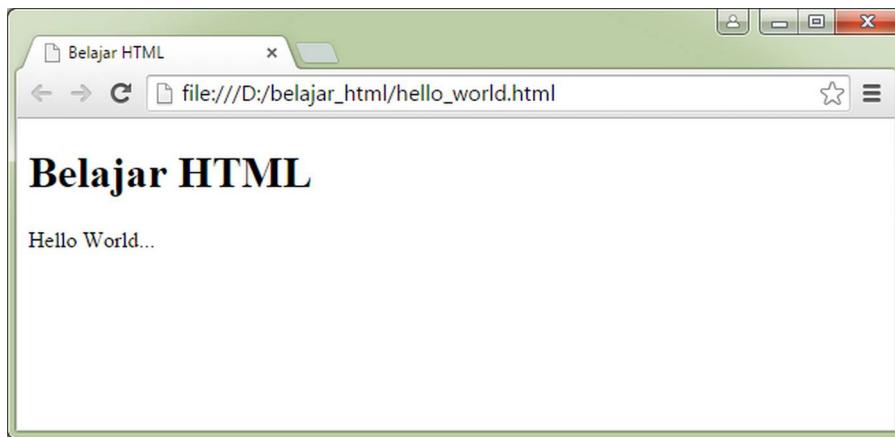
Sesaat setelah anda menyimpan file, kode-kode di dalam Notepad++ akan menjadi berwarna. Hal ini merupakan salah satu fitur standar dari aplikasi text editor khusus programming.

Pewarnaan kode ini dikenal dengan istilah **syntax highlighting**. Ini sangat memudahkan kita dalam membuat perintah HTML. Jika terdapat warna yang tidak pas atau beda, besar kemungkinan ada sesuatu yang salah ketik.



Dalam programming, **syntax** adalah kumpulan aturan penulisan dari suatu bahasa pemrograman. Aturan penulisan ini terdiri dari simbol, karakter, dan kata-kata khusus.

Untuk menjalankan file HTML ini anda bisa langsung buka folder **belajar_html** menggunakan windows explorer, kemudian *double click* file **hello_world.html**. Hasilnya akan ditampilkan ke dalam web browser.



Gambar: Tampilan file hello_world.html



Jika anda mendapati tampilannya berbeda atau bahkan halaman kosong, pastikan kembali kode program sudah sama persis. Dalam pemrograman komputer, satu karakter saja salah, apakah itu tanda titik, koma, atau tanda '>', program tidak akan jalan.

Sebagai alternatif, anda bisa buka web browser terlebih dahulu, misalkan Mozilla Firefox, lalu di bagian address bar silahkan ketik alamat **D:\belajar_html\hello_world.html** dan tekan **Enter**.

Selamat! anda telah berhasil menjalankan file HTML pertama anda!

Dari percobaan ini kita dapat melihat bahwa setiap halaman HTML harus diakhiri dengan ekstensi **.html**. Anda mungkin juga akan menemukan beberapa file HTML yang berakhiran **.htm**. Akhiran ini adalah ekstensi HTML untuk Windows versi lama.

Agar dapat diakses dari web browser, sebuah file HTML harus ditulis tanpa spasi. Sebagai pemisah kata, spasi bisa diganti dengan tanda garis bawah / underscore (_), atau tanda penghubung / hyphen (-).



Halaman HTML yang kita jalankan dari percobaan ini adalah halaman *offline*, yakni tidak terhubung ke internet. Jika anda ingin file HTML tersebut bisa diakses dari internet (berupa sebuah website online), maka harus menyewa **web hosting** dan **web domain**. Lebih lanjut mengenai cara membuat website online ini bisa dibaca di duniaIlkom: [tutorial cara membuat web online²⁰](#).

3.4 Teks Editor Alternatif: Komodo Edit

Salah satu alasan saya menggunakan Text editor **Notepad++** adalah aplikasinya ringan, sangat cepat dan berukuran kecil. Tapi tidak jarang ada yang 'protes' kenapa tampilannya terkesan jadul.

²⁰ <http://www.duniaIlkom.com/tutorial-cara-membuat-website-online/>

Sebagai alternatif, anda bisa mencoba **Komodo Edit**. Teks editor ini juga bisa didapat dengan gratis di: [komodoide.com/komodo-edit²¹](https://www.activestate.com/komodo-ide/downloads/edit). Saat buku ini saya revisi, versi terakhir adalah 11.0.2, berukuran sekitar 75MB.

The screenshot shows the official download page for Komodo Edit. At the top, it says "FREE OPEN SOURCE EDITOR". Below that is a brief description of what Komodo Edit is. To the right are two buttons: "Download Komodo IDE 21 day fully functional trial" (red background) and "Buy Now Choose the right license for you" (dark grey background). Under the description, there's a section titled "OTHER PLATFORMS" with a table showing download links for different operating systems. The table has columns for Version, Windows (x86), Mac OS X (x86_64), Linux (x86), and Linux (x86_64). The row for Version 11.0.2 shows "Windows Installer (MSI)" under Windows (x86), which is highlighted with a red arrow pointing to it. Other options include "Mac Disk Image (DMG)", "AS Package", and "AS Package".

Gambar: Halaman download Komodo Edit



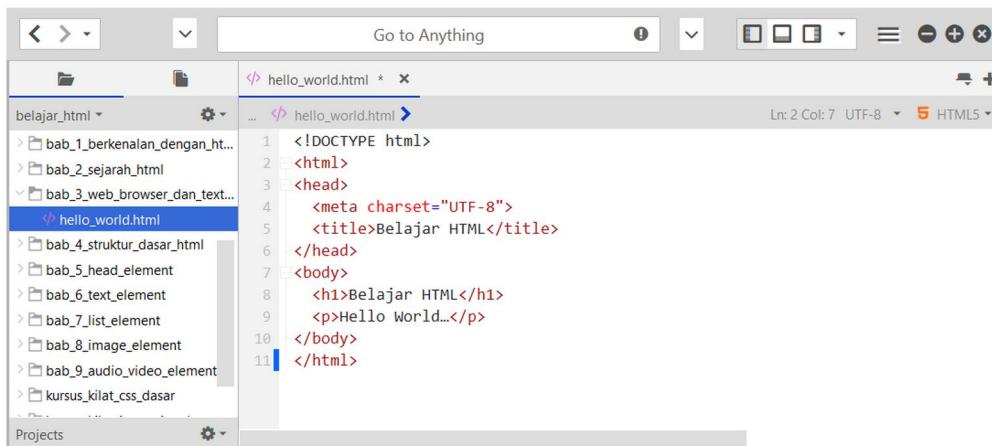
Gambar: Jendela awal instalasi Komodo Edit

Silahkan ikuti proses instalasi seperti biasa. Biarkan seluruh settingan dalam keadaan default.

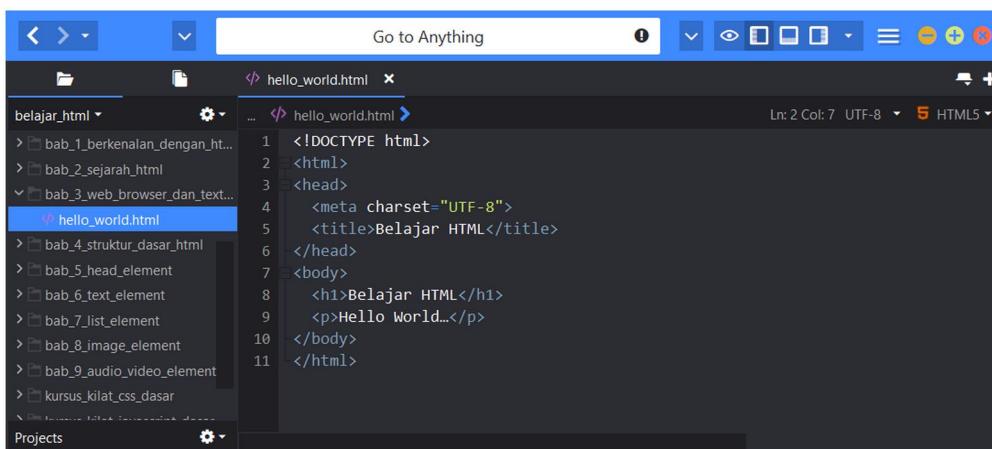
Komodo Edit terlihat lebih modern daripada Notepad++. Jika anda menggunakan Windows 8 atau Windows 10, tampilan jendelanya sangat mirip dengan theme sistem operasi tersebut.

Anda juga bisa mengubah warna background menjadi putih, hitam maupun kombinasi warna lain. Caranya, klik icon hamburger di sisi kanan atas, cari menu Tools -> Color Scheme Editor. Pilih warna theme yang diinginkan dari menu Color Scheme.

²¹<https://www.activestate.com/komodo-ide/downloads/edit>



Gambar: Tampilan Komodo Edit dengan theme putih (classic)



Gambar: Tampilan Komodo Edit dengan theme hitam (default)



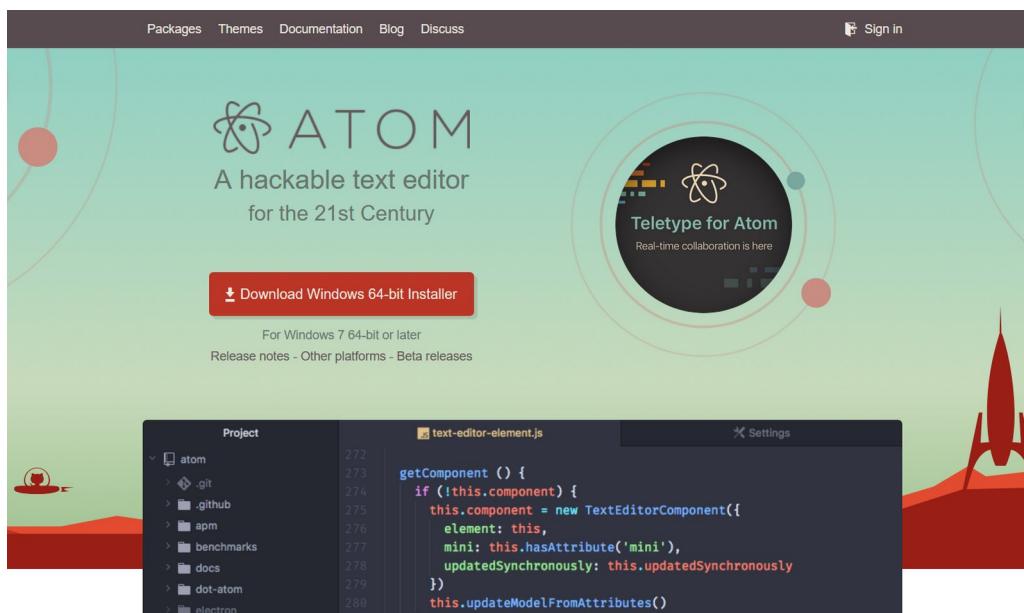
Komodo Edit merupakan versi ringan dari **Komodo IDE**. Karena itulah di situs tersebut anda juga melihat link untuk download Komodo IDE. Aplikasi Komodo IDE merupakan versi berbayar yang fitur-fiturnya tidak kita butuhkan saat ini.

3.5 Teks Editor Alternatif: Atom

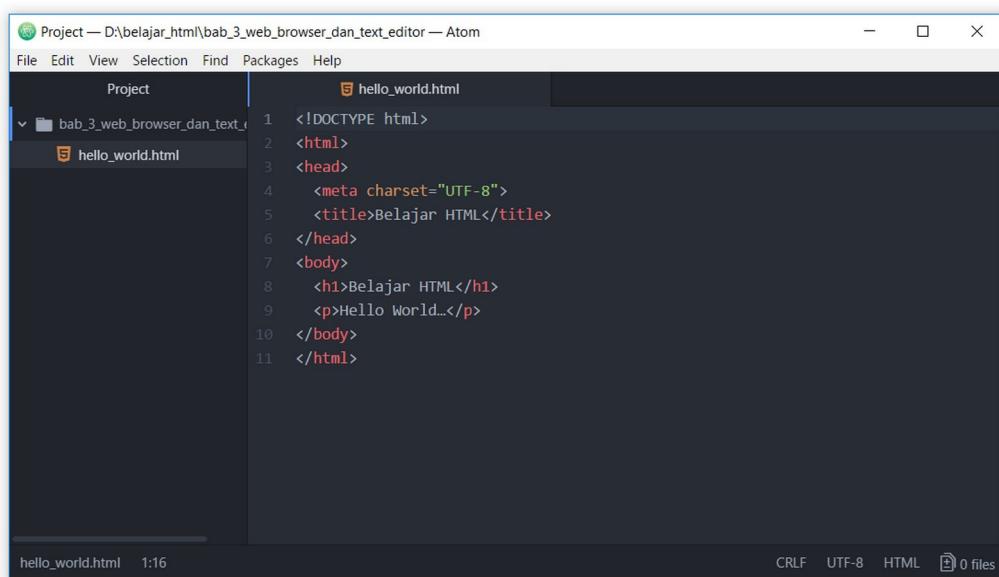
Komodo Edit sebenarnya sangat bagus dan ringan, namun text editor ini tapi tidak terlalu banyak dipakai. Sebenarnya ini lebih ke preferensi masing-masing. Untuk edit kode program yang kecil, saya lebih suka pakai Notepad++ karena ringan dan simple.

Namun jika anda ingin memakai text editor yang lebih populer, saya merekomendasikan **Atom**. Teks editor ini bisa didapat dengan gratis di: [atom.io²²](https://atom.io) dengan ukuran file instalasi sekitar 135 MB. Atom merupakan text editor open source yang didukung oleh **GitHub Inc.**

²²<https://atom.io/>

Gambar: Halaman website Atom di <https://atom.io>

Silahkan ikuti proses instalasi seperti biasa. Biarkan seluruh settingan dalam keadaan default.



Gambar: Membuat kode HTML dengan Atom

Secara bawaan, **Atom** menggunakan tema “gelap” yang menjadi standar editor programming modern. Salah satu alasannya, agar mata kita tidak cepat lelah jika dibandingkan dengan tampilan yang terang (putih). Meskipun demikian, tema tampilan ini nantinya bisa diganti.

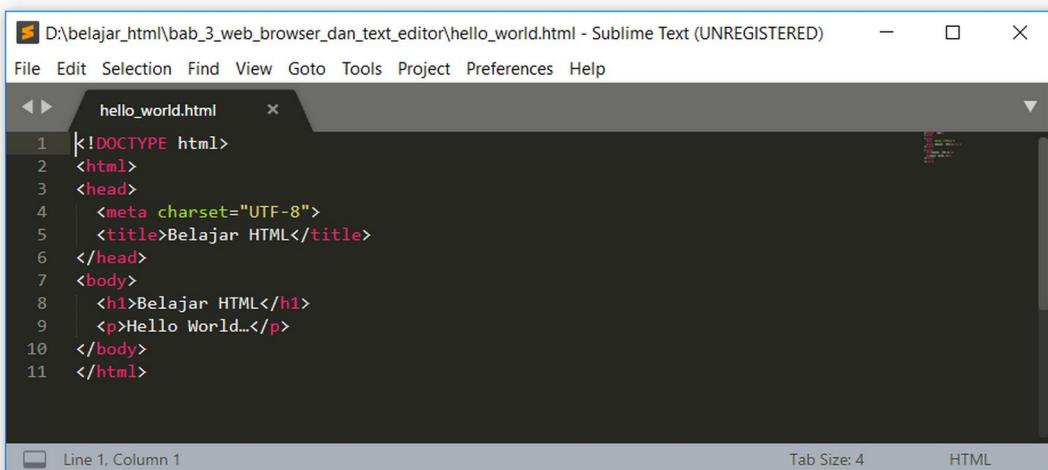
Salah satu keunggulan **Atom** adalah banyaknya plugin yang tersedia (dikenal juga sebagai packages). Package ini ibarat aplikasi tambahan yang berguna untuk menambah fitur-fitur baru ke dalam **Atom**. Misalkan kita banyak membuat kode program menggunakan **Bootstrap** (salah satu framework CSS), tersedia berbagai packages untuk mempermudah pengetikan kode program.

Sebagai informasi tambahan, text editor **Atom** sebenarnya sebuah “web browser khusus”. Aplikasi ini dikembangkan dari **Chromium** dan **Node.js**. Chromium sendiri merupakan sebuah project untuk pengembangan web browser Google Chrome.

3.6 Bagaimana dengan Sublime Text?

Jika anda sudah malang melintang di web programming atau mengikuti berbagai tutorial yang ada di internet, saya yakin banyak yang merekomendasikan text editor **Sublime Text**.

Tidak dapat dipungkiri bahwa **Sublime Text** merupakan text editor yang sangat populer, bahkan mungkin bisa dibilang sebagai text editor yang paling banyak dipakai oleh web programmer professional.



Gambar: Tampilan text editor Sublime Text

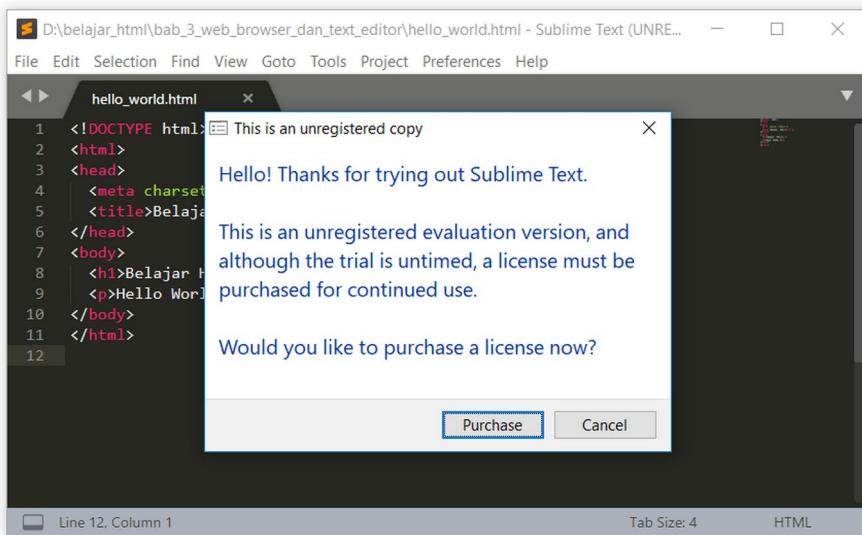
Kenapa saya tidak merekomendasikannya? Alasan satu-satunya adalah karena masalah lisensi. **Sublime Text tidak gratis**, tetapi teks editor berbayar. Harga lisensinya saat ini US \$80, atau sekitar 1 juta rupiah. Terbilang cukup mahal apalagi bagi mahasiswa dan pelajar.

Tapi bukankah Sublime Text bisa di download dengan gratis? Betul, anda bisa mendownload Sublime Text di web resminya di [www.sublimetext.com²³](http://www.sublimetext.com). Akan tetapi ini merupakan versi evaluasi, dimana nanti akan muncul jendela “iklan” secara berkala.

Meskipun tidak ada batas waktu penggunaan, di website sublimetext.com tertulis bahwa “*Sublime Text may be downloaded and evaluated for free, however a license must be purchased for continued use*”.

Kalau diterjemahkan kira-kira menjadi “*Sublime Text bisa di download dan dicoba dengan gratis, tetapi lisensi harus dibeli untuk penggunaan berkali-kali (continued use)*”.

²³[https://www.sublimetext.com](http://www.sublimetext.com)



Gambar: “Iklan” dari Sublime text untuk versi gratis

The screenshot shows the official Sublime Text website. The header features the Sublime Text logo and a 'Download' button. Below the header, there's a large 'Download' button. A sub-section titled 'Version: Build 3143' lists download links for various platforms: OS X (radio button), Windows (radio button), Windows 64 bit (checkbox), and Linux repos (radio button). A note below states: 'Sublime Text may be downloaded and evaluated for free, however a license must be purchased for continued use. There is currently no enforced time limit for the evaluation.' The 'purchased' word is underlined.

Gambar: Lisensi penggunaan versi “gratis” Sublime Text

Daripada menyalahi aturan lisensi (atau yang lebih parah pakai versi bajakan untuk menghilangkan jendela iklan) saya sarankan untuk memakai text editor lain, kecuali anda sanggup membeli lisensi resmi dari Sublime Text.

Berita baiknya, tidak sedikit programmer yang mengatakan bahwa dalam berbagai hal, Atom sudah bisa menyamai Sublime Text. Bahkan ada yang pindah dari Sublime Text ke Atom. Jadi, jika anda saat ini menggunakan Sublime Text bajakan, ada baiknya mencoba teks editor lain yang gratis (agar lebih berkah). Atom merupakan salah satu kandidat terbaik.

Dalam bab kali ini kita telah mempersiapkan perangkat yang dibutuhkan untuk membuat dan menjalankan HTML, yakni web browser dan Text Editor. Sekarang kita bisa memulai mempelajari kode-kode HTML. Bab berikutnya akan membahas mengenai **struktur dasar halaman HTML**.

4. Struktur Dasar HTML

Dalam bab keempat ini akan dibahas tentang cara penulisan HTML. Kita akan mulai dengan mempelajari pengertian tag, element, dan atribut pada HTML, melihat struktur dasar dari kode HTML, mengenal aturan penulisan HTML, serta melakukan validasi kode HTML agar memenuhi standar W3C.

4.1 Pengertian Tag, Atribut dan Element HTML

Tag, atribut dan element adalah 3 bagian inti yang membangun HTML. Mari kita bahas sejenak pengertian ketiga aspek ini.

Pengertian Tag

Dari file `hello_world.html` yang telah kita jalankan dalam bab sebelumnya, tentunya anda sudah mengetahui bahwa file HTML ditulis dengan menggunakan tanda-tanda seperti `<html>`, `<head>`, `<body>` atau `<p>`. Tanda-tanda ini dikenal dengan sebutan HTML tag.

Tag di dalam HTML berfungsi untuk memberi tahu web browser apa dan bagaimana sebuah teks harus ditampilkan. Sebagaimana yang telah kita pelajari, file HTML hanyalah sebuah file text biasa. Web browser lah yang akan memproses file tersebut dan menampilkan hasilnya.

Sebagian besar tag-tag HTML ditulis secara berpasangan, yang terdiri dari tag **pembuka** dan tag **penutup**. Tag pembuka ditulis menggunakan kurung siku, seperti `<p>`. Sedangkan tag penutup ditulis dengan menambahkan tanda garis miring atau *slash* seperti `</p>`.

Tanda `p` disini berarti *paragraf*, dimana kita memberitahukan kepada web browser bahwa seluruh teks yang berada diantara tag pembuka `<p>` dan tag penutup `</p>` adalah **sebuah paragraf**.

Sebagai contoh, jika saya ingin membuat struktur teks yang terdiri dari 2 paragraf, maka saya menulisnya sebagai berikut:

```
<p>Ini adalah paragraf pertama</p>
<p>Ini adalah paragraf kedua </p>
```

Ketika web browser membaca kode diatas, teks diantara tag `<p>` dan tag `</p>` akan ditampilkan sebagai paragraf yang terpisah.

Setiap tag memiliki fungsi dan peran masing-masing. Total, terdapat sekitar seratusan tag di dalam HTML. Walaupun HTML memiliki banyak tag, tidak semuanya harus digunakan. Sebagai contoh, tag `<p>` akan sangat sering kita jumpai, tetapi tag `<wbr>` relatif jarang ditemukan.

Dalam buku ini saya akan membahas sebagian besar tag-tag yang ada di dalam HTML, namun kita akan fokus kepada tag-tag yang sering digunakan dan memiliki peran penting untuk membuat struktur halaman.

Dalam menjalankan tugasnya, tag HTML memerlukan **atribut**.

Pengertian Atribut

Atribut adalah informasi tambahan yang ditulis pada **tag pembuka**. Fungsi dari atribut ini bermacam-macam tergantung nilai dan pada tag mana ia ditempatkan. Beberapa atribut bersifat umum dan bisa digunakan dalam seluruh tag (dikenal sebagai *global atribut*), tetapi kebanyakan hanya berfungsi untuk tag tertentu saja.

Atribut terdiri dari pasangan **nama atribut** dan **nilai atribut**. Sebagai contoh, untuk menambahkan atribut **class** dengan nilai **pertama** ke dalam tag **<p>**, cara penulisannya adalah sebagai berikut:

```
<p class="pertama">Ini adalah paragraf pertama</p>
```

Penulisan nilai dari atribut boleh menggunakan **tanda kutip dua** (") seperti diatas, atau boleh juga menggunakan **tanda kutip satu** (') seperti contoh berikut:

```
<p class='pertama'>Ini adalah paragraf pertama</p>
```

Dalam HTML, penulisan nilai atribut diantara tanda kutip sebenarnya bersifat *opsional* (boleh tidak ditulis). Contoh diatas juga bisa dibuat sebagai berikut:

```
<p class=pertama>Ini adalah paragraf pertama</p>
```

Penulisan nilai atribut tanpa tanda kutip seperti ini tidak salah dan tetap valid di dalam HTML. Akan tetapi, jika nilai atribut terdiri dari 2 kata atau lebih, maka kita harus menggunakan tanda kutip, seperti contoh berikut:

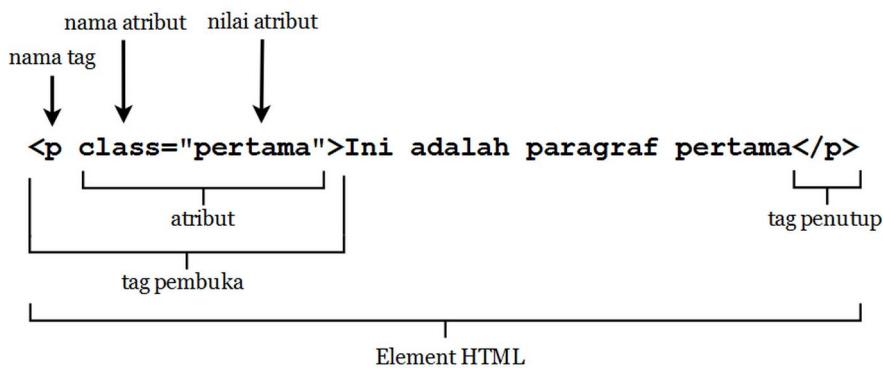
```
<p class="pertama penting">Ini adalah paragraf pertama dan penting</p>
```



Penggunaan tanda kutip untuk nilai atribut berasal dari format **XHTML**, dimana setiap atribut harus ditutup dengan tanda kutip. HTML5 tidak mengharuskan cara penulisan tanda kutip ini. Walaupun demikian, kebanyakan programmer tetap menambahkannya.

Pengertian Element

Istilah ketiga yang masih berkaitan dengan **tag** dan **atribut** adalah **element**. Di dalam HTML, **element** adalah satu bagian utuh yang terdiri dari **tag**, **atribut**, dan seluruh teks yang berada di antara tag pembuka dan penutup. Agar lebih jelas, konsep dari element ini dapat anda lihat dalam gambar dibawah ini.



Gambar: Bagian tag, atribut dan element dari HTML

Dari gambar diatas kita dapat melihat bahwa element mencakup tag, atribut dan seluruh isi dari tag tersebut, termasuk jika didalamnya juga terdapat tag lain.



Dalam pembahasan HTML, istilah **tag** dan **element** kadang saling dipertukarkan. Beberapa referensi lebih banyak menyebut **tag** <p>, tetapi ada pula yang menyebutnya sebagai **element** p. Dalam buku ini saya akan menggunakan keduanya secara bergantian. Untuk penulisan judul, saya cenderung menggunakan istilah *element*, dan menggunakan istilah *tag* dalam pembahasan.

Tag, atribut dan **element** adalah inti dari HTML. Kita akan mempelajarinya sepanjang buku ini.

4.2 Struktur Dasar HTML

File **hello_world.html** yang kita jalankan sebelum ini sebenarnya telah memenuhi struktur dasar dari sebuah file HTML. Mari kita lihat kembali kode HTML tersebut:

hello_world.html

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4    <meta charset="UTF-8">
5    <title>Belajar HTML</title>
6  </head>
7  <body>
8    <h1>Belajar HTML</h1>
9    <p>Hello World...</p>
10 </body>
11 </html>
```

Jika anda perhatikan, sebagian besar tag-tag diatas saling berpasangan, kecuali tag <!DOCTYPE html> dan <meta charset="UTF-8">. Beberapa tag juga berada di dalam tag lain. Mari kita bahas mulai dari baris pertama.

DTD: Document Type Declaration

Pada baris pertama file `hello_world.html`, terdapat tag `<!DOCTYPE html>`. Tag khusus ini dikenal dengan sebutan DTD (*Document Type Declaration*) atau `doctype`. DTD berfungsi untuk menginformasikan web browser tentang aturan penulisan dari suatu dokumen.

DTD berasal dari sistem SGML (bahasa markup dimana HTML berasal). Di dalam SGML, setiap dokumen harus berisi penjelasan mengenai tipe dan jenisnya, dan harus ditulis pada baris pertama.

Sebelum HTML5, penulisan DTD di dalam HTML sangat panjang dan hampir mustahil untuk dihafal. Sebagai contoh, berikut adalah DTD untuk HTML 4.01:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
"http://www.w3.org/TR/html4/strict.dtd">
```

HTML5 mencoba mengubah hal ini agar lebih praktis. Pengembang HTML5 (organisasi WHATWG) memutuskan bahwa HTML5 akan sepenuhnya terlepas dari aturan SMGL. Dengan kata lain, HTML5 tidak lagi bagian dari SMGL, dan merupakan bahasa markup yang sepenuhnya berdiri sendiri. Oleh karena itu, penulisan DTD sebenarnya tidak lagi diperlukan.

Akan tetapi, penghapusan DTD dari HTML terkendala dengan sebuah mekanisme di dalam web browser yang dinamakan **quirks mode**.

Quirks mode adalah sebuah model tampilan yang digunakan web browser ketika menampilkan kode HTML yang tidak standar. Umumnya, halaman HTML yang tidak standar ini berasal dari situs website ‘tua’ yang dibuat puluhan tahun lalu.

Untuk membedakan apakah sebuah halaman web termasuk standar atau tidak, web browser akan melihat apakah di dalam dokumen terdapat DTD atau tidak. Jika tidak ditemukan, web browser akan tampil dalam *quirks mode*.

Ketika sebuah dokumen di proses dalam *quirks mode*, web browser akan membuat beberapa perlakuan khusus. Misalnya, menampilkan web dengan tampilan yang berbeda. Ini akan menjadi masalah karena desain web yang telah dirancang bisa jadi berantakan. Sebagai *web developer*, sedapat mungkin kita harus menghindari hal ini.

Pihak WHATWG kemudian mencari solusi dengan membuat DTD sesingkat mungkin agar web browser tidak masuk ke dalam *quirks mode*. Caranya adalah dengan menggunakan penulisan DTD: `<!DOCTYPE html>`. Dengan demikian, web browser tetap menggunakan mode normal yang dinamakan **Standards Compliance Mode**.

Posisi DTD juga harus ditulis pada baris paling awal. Beberapa web browser (terutama IE versi awal) akan masuk ke dalam quirks mode jika ditemukan ada karakter lain sebelum penulisan DTD (walaupun itu adalah sebuah spasi).

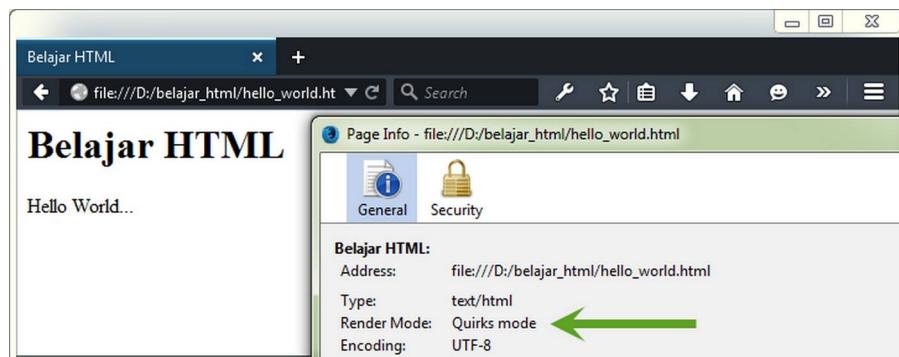


Penulisan DTD dalam HTML5 boleh ditulis dengan huruf kecil, seperti `<!doctype html>`. Namun anda akan sering menemukan penulisannya yang menggunakan huruf besar, seperti: `<!DOCTYPE html>`.

Mencoba Quirks Mode dan Standards Compliance Mode

Jika anda ingin melihat perbedaan antara *quirks mode* dan *standards compliance mode*, silahkan buat 2 buah file HTML (yang salah satunya tanpa menggunakan DTD). Hal ini lebih mudah jika diakses menggunakan Mozilla Firefox.

Pertama, jalankan file yang ingin diperiksa, misalkan `hello_world.html`. Setelah halaman tampil di web browser, klik kanan halaman web dan pilih **View Page Info**. Dalam jendela **File Info**, pilih tab **General** dan cari baris **Render Mode**. Anda akan melihat apakah halaman web diproses dalam *quirks mode* atau *standards compliance mode*.



Gambar: Halaman ini diproses dalam Quirks Mode

Tag <html>

Setelah penulisan DTD, struktur halaman HTML kemudian diikuti dengan tag `<html>`. Dapat anda perhatikan bahwa tag `<html>` akan ‘*membungkus*’ seluruh kode-kode HTML lain.

Tag pembuka `<html>` berada di baris paling atas setelah `<!DOCTYPE html>`, dan tag penutup `</html>` berada di baris paling akhir dari kode HTML. Seluruh tag HTML lain harus berada di dalam tag ini.

Di dalam tag `<html>`, bisa diletakkan atribut `lang`. Atribut ini diisi dengan dua digit kode bahasa yang akan digunakan untuk isi website. Sebagai contoh, jika kita membuat website dengan bahasa indonesia, bisa menulis tag `<html>` sebagai: `<html lang="id">`. Atau `<html lang="en">` jika akan membuat website dengan bahasa inggris.

Atribut `lang` sepenuhnya opsional (boleh ditulis, boleh juga tidak). Namun anda akan sering menemukan atribut ini. Daftar lengkap tentang dua digit kode negara bisa dilihat di [wikipedia¹](http://en.wikipedia.org/wiki/List_of_ISO_639-1_codes).



Struktur HTML yang ada di dalam file `hello_world.html` akan kembali saya bahas secara detail di dalam bab-bab berikutnya. Disini kita hanya membahas sekilas agar anda dapat memahami sebuah dokumen HTML lengkap.

¹http://en.wikipedia.org/wiki/List_of_ISO_639-1_codes

Tag <head>

Tag <head> berfungsi sebagai ‘penampung’ berbagai tag yang umumnya tidak akan terlihat dalam web browser. Tag-tag di dalam bagian <head> berfungsi untuk memberikan informasi tambahan mengenai halaman yang sedang diproses, atau menyediakan referensi ke file lain (seperti file CSS atau JavaScript).

Beberapa tag HTML lain yang biasa terdapat pada bagian <head> adalah: <meta>, <title>, dan <link>.

Tag <meta>

Tag <meta> merupakan singkatan dari *metadata*. **Metadata** adalah data yang berisi informasi tentang suatu dokumen. Tag <meta> berisi informasi/instruksi mengenai halaman HTML yang sedang dijalankan.

Salah satu tag <meta> yang terdapat dalam struktur diatas adalah <meta charset="UTF-8">. Tag ini akan menginstruksikan web browser untuk menggunakan karakter set UTF-8 dalam memproses halaman HTML.

Tag <title>

Tag <title> merupakan satu dari sedikit tag yang bisa terlihat di web browser yang berada pada bagian <head>.

Sesuai dengan namanya, tag ini berfungsi untuk membuat *title* dari sebuah halaman. **Title** adalah teks yang ditampilkan pada bagian atas jendela web browser. Selain itu, jika kita men-bookmark sebuah halaman, title inilah yang akan menjadi nama bookmark tersebut.

Tag <body>

Tag <body> berfungsi sebagai penampung (*container*) dari seluruh kode HTML yang tampil di dalam web browser. Disinilah terletak 90% tag-tag HTML yang akan kita tulis nantinya.

Tag <h1> dan <p>

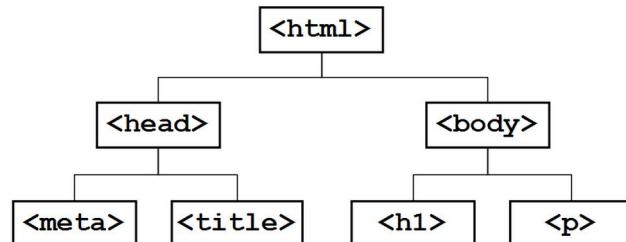
Tag <h1> merupakan singkatan dari “header1”, yang digunakan untuk membuat struktur judul di dalam HTML. HTML menyediakan 6 tipe header, mulai dari <h1>, <h2>, sampai dengan <h6>.

Tag <p> adalah singkatan dari “paragraf” yang digunakan untuk...(yah) membuat paragraf. Saya akan membahas tag <h1> dan tag <p> dengan lebih detail dalam bab selanjutnya.

4.3 Mengenal HTML Tree / Document Object Model (DOM)

Jika diperhatikan, tag-tag yang ada pada HTML memiliki struktur tersendiri. Struktur ini biasa disebut **HTML tree** (pohon HTML) atau istilah kerennya **Document Object Model (DOM)**.

Sesuai dengan istilah tersebut, kode-kode HTML dapat diibaratkan seperti pohon, dimana setiap tag saling terhubung satu sama lain. Agar lebih mudah dipahami, perhatikan gambar diagram dibawah ini.



Gambar: Struktur DOM halaman hello_world.html

Diagram diatas adalah struktur DOM dari kode HTML halaman **hello_world.html** yang kita bahas sebelumnya. Tag paling atas, yakni tag `<html>` disebut sebagai tag “**akar**” (**root**) karena seluruh kode HTML lain harus berada di dalam tag ini. Konsep struktur DOM ini sangat penting untuk dipahami, terutama sebagai dasar ketika mempelajari **JavaScript**.

Selain sebagai “pohon”, diagram diatas juga bisa dibaca seperti diagram keluarga. Sebagai contoh, tag `<head>` dan tag `<body>` berada persis dibawah tag `<html>`. Oleh karena itu, tag `<head>` dan tag `<body>` disebut sebagai “**anak**” (**child**) dari tag `<html>`, dan tag `<html>` disebut sebagai “**induk**” (**parent**). Begitu juga dengan tag `<h1>` dan `<p>` yang merupakan anak dari tag `<body>`.

Contoh diagram DOM yang kita bahas disini adalah versi sederhana dari kode HTML yang sebenarnya. Untuk halaman yang lebih kompleks, diagram diatas bisa mencapai 10 tingkatan atau lebih.

4.4 Aturan Dasar Penulisan HTML

Setelah membahas tentang struktur dasar HTML, kita akan masuk ke dalam aturan dasar penulisan HTML. Karena kita menggunakan HTML5, maka aturannya sedikit berbeda bila dibandingkan dengan XHTML (terutama jika anda pernah mempelajarinya).

XHTML adalah ‘pendahulu’ dari HTML5. XHTML pada awalnya dirancang sebagai penerus HTML 4.01 yang dianggap ‘terlalu bebas’, dan karena itu XHTML dikenal dengan aturan penulisan yang lebih ketat.

Ketika pengembangan XHTML berhenti dan tidak dilanjutkan, HTML5 hadir sebagai penerusnya. HTML5 membuat aturan-aturan penulisan ini menjadi ‘kembali longgar’. Sebagai contoh, jika di dalam XHTML penulisan *tag* dan *atribut* harus menggunakan huruf kecil, di dalam HTML5 hal ini tidak menjadi masalah.

Case Sensitivity

Case Sensitivity adalah istilah yang membahas apakah sebuah bahasa pemrograman membedakan penulisan huruf kecil dan huruf besar.

Di dalam XHTML, seluruh tag, nama atribut, dan nilai atribut harus ditulis di dalam huruf kecil. Akan tetapi dalam HTML5, penulisan huruf besar dan huruf kecil akan dianggap sama. Baik tag `<P>` maupun tag `<p>` sama-sama diperbolehkan di dalam HTML5.

Walaupun demikian, karena faktor kebiasaan dan agar terkesan lebih rapi, saya menyarankan anda untuk menggunakan huruf kecil pada penulisan seluruh tag, atribut, dan nilai atribut.

Pengecualiannya adalah pada penulisan DTD, yakni ditulis menggunakan huruf besar: `<!DOCTYPE html>`. Walaupun kita juga bisa menulisnya menggunakan huruf kecil, seperti `<!doctype html>`, tapi mayoritas *web programmer* menggunakan huruf besar untuk penulisan DTD.

Self Closing Tag

Beberapa tag di dalam HTML ada yang tidak memiliki pasangan tag pembuka dan tag penutup, contohnya adalah tag `<meta>`, `
` dan ``. Tag-tag ini disebut juga sebagai **void element**.

Di dalam XHTML, tag yang berdiri sendiri harus ditutup dengan cara menambahkan tanda garis miring depan ‘ / ‘ (*forward slash*) sebelum kurung siku penutup tag.

Sebagai contoh, tag `
` yang digunakan untuk membuat baris baru, di dalam XHTML ditulis menjadi `
`. Penulisan seperti ini disebut juga dengan **self closing tag**.

Di dalam HTML5, aturan ini tidak lagi diwajibkan. Penulisan `
` maupun `
` sama-sama dianggap “sah”.

Beberapa programmer ada yang tetap menggunakan ‘gaya penulisan’ XHTML seperti: `
` karena terkesan lebih rapi. Tapi ada pula yang berpendapat bahwa ini adalah cara ‘kuno’, dan cukup menulisnya dengan: `
`. Anda boleh mengikuti cara penulisan yang manapun, asal konsisten sepanjang kode HTML agar tidak membuat bingung.

Pasangan Tag Penutup

Karena sebagian besar tag ditulis berpasangan, kita harus hati-hati agar tidak lupa menulis pasangan tag penutup-nya. Kesalahan ini bisa berakibat tampilan halaman web jadi berantakan.

Sebagai contoh, tag `` dan tag `` digunakan untuk mempertegas sebuah kata atau kalimat. Tag `` akan ditampilkan dengan huruf miring (*italic*), dan tag `` akan ditampilkan dengan huruf tebal (*bold*). Kedua tag ini banyak dipakai dalam paragraf.

Jika kita ‘lupa’ menutup tag `` atau tag `` tampilan seluruh halaman bisa jadi rusak. Sebagai contoh, perhatikan kode HTML berikut ini:

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar HTML</title>
6 </head>
7 <body>
8   <h1>Belajar HTML</h1>
9   <p>Salah satu hal yang unik dari <strong>HTML</strong>
10    (dan juga <em>CSS & JavaScript</em>) adalah kita bisa
11     melihat dengan bebas kode-kode yang digunakan untuk
12     membuat sebuah halaman web.</p>
13 </body>
14 </html>

```



Gambar: Lupa menutup tag

Jika anda menjalankan kode HTML diatas, seluruh teks mulai dari ‘CSS & JavaScript’ hingga ‘halaman web.’ ditampilkan dengan huruf miring. Padahal yang saya inginkan untuk tampil miring hanya teks ‘CSS & JavaScript’ saja.

Letak kesalahan pada kode diatas adalah pada tag , dimana saya lupa untuk menambahkan tanda penutup “”. Seharusnya halaman diatas ditulis sebagai berikut:

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar HTML</title>
6 </head>
7 <body>
8   <h1>Belajar HTML</h1>
9   <p>Salah satu hal yang unik dari <strong>HTML</strong>
10    (dan juga <em>CSS & JavaScript</em>) adalah kita bisa
11     melihat dengan bebas kode-kode yang digunakan untuk
12     membuat sebuah halaman web.</p>
13 </body>
14 </html>

```



Gambar: Tag `` diakhiri dengan penutup ``

Kali ini tampilan akhir sesuai dengan yang kita inginkan.

Whitespace

Whitespace adalah istilah yang merujuk kepada karakter ‘*spasi*’ yang tidak tampil di layar, contohnya adalah karakter spasi, tab, dan karakter **enter** (yang dikenal juga dengan karakter **carriage returns**).

Di dalam HTML, *whitespace* akan diabaikan, termasuk jika ditulis di dalam teks. Sebagai contoh, kedua kode HTML dibawah ini akan menghasilkan tampilan yang sama:

```

1  <!DOCTYPE html>
2  <html><head><meta charset="UTF-8"><title>Belajar HTML</title>
3  </head><body><h1>Belajar HTML</h1><p>Jangan diganggu! lagi serius belajar.</p>
4  </body></html>

```

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Belajar HTML</title>
6  </head>
7  <body>
8      <h1>Belajar HTML</h1>
9      <p>Jangan diganggu! lagi serius belajar.</p>
10 </body>
11 </html>

```

Walaupun whitespace tidak akan diproses, menyisipkan *spasi* dan *tab* ke dalam kode HTML akan membuatnya lebih mudah dibaca.

Berkaitan dengan **whitespace**, dalam HTML terdapat tag `
` yang bisa digunakan sebagai pengganti ‘Enter’. Tag `
` merupakan singkatan dari **break**, akan membuat teks setelahnya tampil pada baris baru. Tag ini merupakan salah satu tag yang sangat sering anda jumpai.

Selain itu, HTML juga memiliki tag `<hr>` (Horizontal Line) untuk membuat garis horizontal. Berikut contoh penggunaannya:

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar HTML</title>
6 </head>
7 <body>
8   Berikut adalah beberapa aplikasi web browser yang bisa anda gunakan: <br>
9   <hr>
10  Google Chrome <br>
11  Mozilla Firefox <br>
12  Microsoft Internet Explorer <br>
13  Opera <br>
14  Apple Safari <br>
15 </body>
16 </html>

```



Gambar: Tampilan tag
 dan <hr>



Contoh diatas sebenarnya lebih cocok jika menggunakan *list*, kita akan mempelajari tentang **List Element** dalam bab tersendiri nantinya.

Membuat Komentar

Dalam membuat kode HTML (dan juga kode program pada umumnya), kadang kita perlu menambahkan beberapa keterangan di dalam kode program, apakah itu berupa catatan kecil tentang tanggal dan waktu program dibuat, atau keterangan untuk apa kita menulis kode tersebut.

Keterangan ini biasa disebut **komentar** (*comment*). Di dalam HTML, komentar dibuat menggunakan tag pembuka: `<!--` dan tag penutup: `-->`. Seluruh teks yang ada diantara kedua tag ini akan dianggap sebagai komentar sehingga tidak akan diproses dan tidak ditampilkan oleh web browser.

Berikut adalah contoh penulisan komentar di dalam HTML:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar HTML</title>
6 </head>
7 <body>
8   <!-- Ini adalah komentar, dan tidak akan ditampilkan pada web browser -->
9   <h1>Belajar HTML</h1>
10  <p>HTML is fun!</p>
11 </body>
12 </html>
```

Selain sebagai catatan, komentar juga sering digunakan untuk membuat keterangan siapa pembuat kode program, atau membatasi sebuah blok kode HTML. Berikut contoh penggunaannya:

```
1 <!DOCTYPE html>
2 <!--
3 Author: Andre
4 Author URL: http://www.duniaIlkom.com
5 Version: 1.3
6 License: GNU General Public License v2 or later
7 -->
8 <html>
9 <!-- start head section -->
10 <head>
11   <meta charset="UTF-8">
12   <title>Belajar HTML</title>
13 </head>
14 <!-- start body section -->
15 <body>
16   <!-- main article -->
17   <h1>Belajar HTML</h1>
18   <p>HTML is fun!</p>
19 </body>
20 </html>
```



Gambar: Bagian komentar tidak akan ditampilkan

Dalam contoh kode HTML diatas, saya menggunakan beberapa komentar untuk membuat identitas file HTML serta menandai bagian-bagian tertentu dari halaman. Metode penulisan seperti ini akan memudahkan pembacaan struktur halaman terutama untuk kode HTML yang panjang.



Perhatikan bahwa untuk membuat tanda pembuka komentar, kita membutuhkan 5 karakter, yakni: <, !, -, -, , dan sebuah spasi. Untuk menutup komentar, gunakan 4 karakter: spasi, -, -, dan >. Karakter spasi ini sering diabaikan, tetapi beberapa web browser membutuhkan hal ini.

4.5 Standar & Validasi HTML

Sepanjang kita membuat halaman web, sedapat mungkin agar mengikuti aturan penulisan HTML yang sesuai standar W3C. Jika sebuah halaman web dapat tampil baik, belum tentu hal tersebut tetap berlaku beberapa tahun ke depan. Sebuah halaman web yang mengikuti standar disebut juga dengan istilah **standards-compliant**.

Di dalam pemrograman web, **standard** merujuk kepada dokumen resmi tentang cara penulisan dan cara penggunaan tag dan/atau atribut. Karena kita akan menggunakan HTML5, maka standar yang akan kita gunakan harus sesuai dengan apa yang dirancang W3C.

Anda bisa mempelajari standar HTML5 di <http://www.w3.org/TR/html5/>². Tetapi dokumen ini sangat teknis, dan agar bisa memahaminya sebaiknya sudah paham tentang HTML secara umum.

Mengikuti standar W3C akan membuat kode kita menjadi ‘**future-proof**’, yakni tampilan dan struktur website kita tetap akan relevan di masa mendatang.

Walaupun sudah terdapat standar baku mengenai tag dan atribut, web browser kadang juga menambahkan tag dan atribut versi mereka sendiri. Tag dan atribut ini disebut sebagai tag dan atribut **non-standar**.

Sebagai contoh, tag `<blink>` dan `<marquee>` merupakan tag non-standar yang hadir pada era ‘browser war’ antara *Netscape* dan *Internet Explorer*. Karena bersifat non-standar, terdapat beberapa web browser yang tidak mendukung kedua tag ini.

Tag `<blink>` akan diabaikan oleh Google Chrome dan Mozilla Firefox, sedangkan tag `<marquee>` masih di dukung oleh kebanyakan web browser.

²<http://www.w3.org/TR/html5/>

Cara Memeriksa Validasi HTML

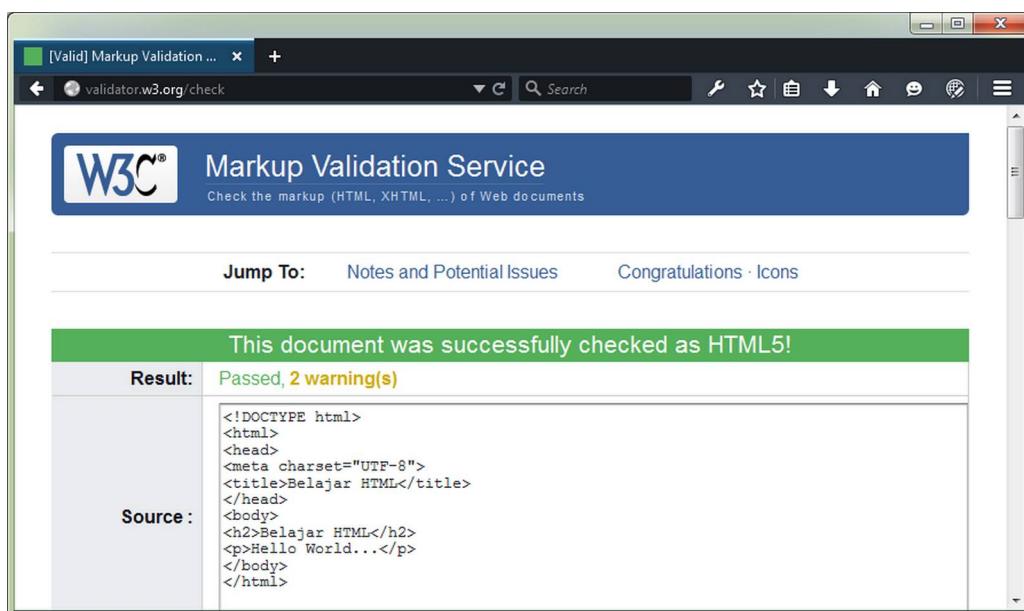
Untuk memeriksa apakah kode-kode HTML yang kita buat sudah sesuai dengan standar atau tidak, bisa mengunjungi situs <http://validator.w3.org>³. Situs ini dikelola langsung oleh W3C sebagai badan yang membuat standar baku HTML.

Situs **validator.w3.org** menyediakan 3 pilihan untuk mengecek validasi: menggunakan alamat situs, men-upload file HTML, atau dengan men-copy paste kode HTML (*direct input*).

Kali ini kita akan coba untuk memvalidasi kode HTML yang ada pada halaman **hello_world.html**:

1. Buka situs <http://validator.w3.org>⁴.
2. Dari pilihan validasi, pilih **Validate by Direct Input**.
3. Copy-paste kode HTML yang ada di dalam file **hello_world.html** ke dalam kotak inputan.
4. Klik tombol **Check**

Hasil dari validasi tersebut terlihat seperti pada gambar dibawah. Dan, hasilnya adalah: *This document was successfully checked as HTML5!*, yang berarti valid sebagai HTML5.



Gambar: Hasil validasi kode HTML file hello_world.html

Jika anda bertanya kenapa ada informasi *Result: Passed, 2 warning(s)*, warning (peringatan) ini muncul karena hal berikut:

1. Validasi HTML5 masih dalam tahap pengembangan, sehingga situs validator.w3.org memberi peringatan bahwa mungkin terdapat beberapa informasi yang kurang valid.

³<http://validator.w3.org>

⁴<http://validator.w3.org>

2. Karena kita menggunakan direct input, proses validasi karakter *encoding* tidak bisa dilakukan. Jika kita memilih untuk *upload* file html (bukan dengan cara copy-paste kode program), pesan warning ini tidak akan muncul. Ini karena kita telah menyertakan tag <meta charset="UTF-8">.

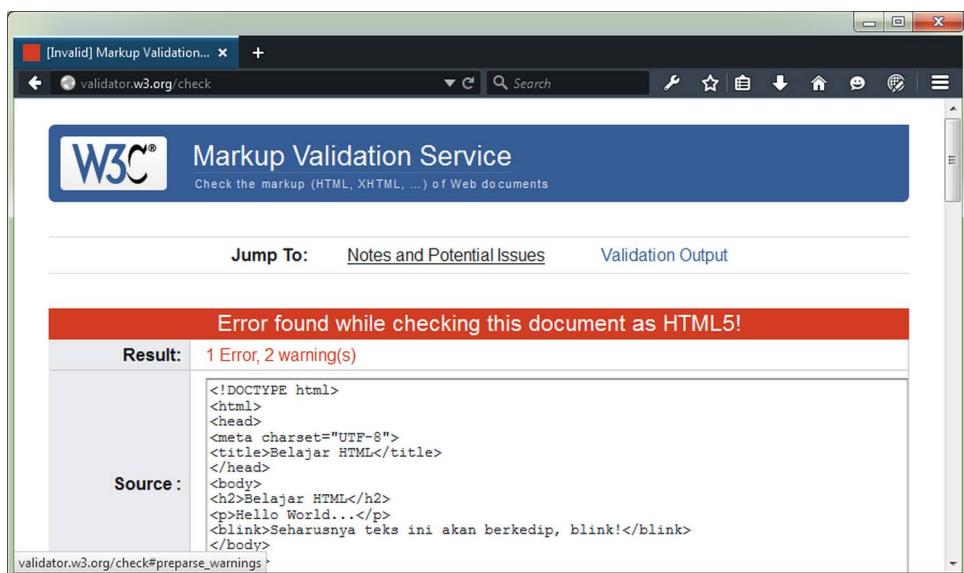
Sebagai percobaan kedua, kali ini tambahkan tag <blink> di dalam kode HTML setelah tag <p>, seperti berikut ini:

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Belajar HTML</title>
6  </head>
7  <body>
8      <h2>Belajar HTML</h2>
9      <p>Hello World...</p>
10     <blink>Seharusnya teks ini akan berkedip, blink!</blink>
11  </body>
12 </html>
```

Tag <blink> merupakan tag non-standar yang dibuat oleh programmer web browser *Netscape* pada tahun 2000-an. Tag ini akan membuat teks dengan efek kedap-kedip (hilang timbul) yang cukup fenomenal (pada saat itu).

Jika anda memvalidasi kode diatas, maka hasilnya adalah: *Error found while checking this document as HTML5!*, seperti pada gambar dibawah.



Gambar: Hasil validasi tag blink

Error ini muncul karena tag <blink> merupakan tag non-standar dan tidak valid di dalam HTML5.



Sebuah hal menarik ketika saya menulis tentang tag <blink> ini. Google ternyata memiliki sebuah **easter egg** unik (kerjaan iseng programmer di google). Silahkan anda ketik di google search: "blink tag", dan hasil pencarian akan berkedap-kedip!

Haruskah Lulus Validasi?

Dari penjelasan sebelumnya, saya menyarankan anda untuk sedapat mungkin merancang kode HTML yang lulus validasi, atau *standards-compliant*. Namun dunia nyata tak seindah itu (eh...)

Mari kita kembali ke halaman validator.w3.org⁵, kali ini pilih **Validate by URI**. Kita akan mencoba *test-drive* beberapa website yang (menurut saya) seharusnya lulus validasi. Mari kita coba situs google. Pada kolom **Address** inputlah: www.google.com, kemudian klik **Check**. Bagaimana hasilnya?

Pada saat saya mencoba ini, hasilnya adalah: **28 Errors, 5 warning(s)**. Mari kita coba situs [yahoo.com](http://www.yahoo.com), dan saya mendapat **268 Errors, 8 warning(s)!**

Meskipun sebagian besar referensi mewajibkan kita untuk taat kepada standar, namun dalam kenyataannya hal ini sangat sulit dilakukan. Terutama jika situs web anda sudah sedemikian kompleks dan melibatkan CSS + JavaScript. Situs duniailkom.com pun juga tidak lulus validasi (ketika saya coba, hasilnya terdapat **21 Errors, 11 warning(s)**).

Satu hal yang menjadi catatan adalah: web browser tidak terlalu peduli apakah sebuah website lulus validasi atau tidak. Web browser dirancang untuk menampilkan website dengan 'sebagus mungkin', walaupun pada halaman tersebut terdapat beberapa tag yang tidak sesuai standar atau tidak sesuai dengan spesifikasi 'resmi'.

Beberapa programmer juga ada yang berpendapat bahwa selama website tersebut tampil bagus pada mayoritas web browser, maka 1 atau 5 kesalahan validasi tidak menjadi masalah. Dengan catatan, kesalahan validasi tersebut bukan hal fatal, misalnya penggunaan atribut non-standar untuk meta-tag. Hal ini akan berakibat gagal validasi, tetapi tidak berdampak serius.

Walaupun demikian, saya tetap menyarankan untuk sedapat mungkin membuat website yang lulus validasi. Namun jika hal itu membutuhkan waktu yang tidak sebentar (terutama jika anda sudah membuat ribuan baris program, dan harus mengulang kembali), then, just let it go...

4.6 Block Level Element dan Inline Level Element

Sebelum saya menutup bab ini, mari kita bahas sedikit tentang **block level element** dan **inline level element**.

Jika dilihat dari cara web browser menampilkan sebuah element (atau tag), terdapat 2 jenis tampilan, yakni **block** dan **inline**.

Block level element (atau *block level tag*) adalah tag-tag HTML yang menjadi bagian terpisah dari alur halaman, dan ditampilkan dalam baris baru. Contoh dari block level element adalah

⁵ <http://validator.w3.org>

tag <p> dan tag <h1>. Kedua tag ini akan ‘menutup’ tag sebelumnya dan memulai sebuah baris baru (memulai block baru). Umumnya tag ini berada di bagian paling luar dari struktur DOM.

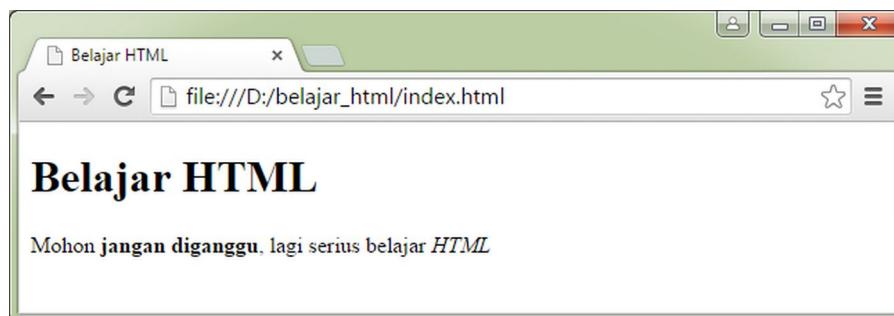
Inline level element (atau *inline level tag*) adalah tag-tag HTML yang mengikuti tampilan yang sudah ada. Tag ini tidak akan membuat baris baru. Contoh dari inline level element adalah tag dan . Kedua tag ini digunakan untuk membuat huruf miring dan huruf tebal pada sebuah teks.

Inline level element umumnya tidak berdiri sendiri tetapi berada di dalam block level element.

Sebagai contoh, dan juga sebagai *template* untuk kode-kode HTML yang akan kita gunakan dalam bab berikutnya, save lah kode HTML berikut dengan nama **index.html**:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar HTML</title>
6 </head>
7 <body>
8   <h1>Belajar HTML</h1>
9   <p>Mohon <strong>jangan diganggu</strong>,
10  lagi serius belajar <em>HTML</em></p>
11 </body>
12 </html>
```



Gambar: Tampilan index.html

Dari tampilan diatas, anda dapat melihat perbedaan dari **block level element** dengan **inline level element**.

index.html

File dengan nama “**index**” memiliki peran khusus di dalam HTML. File ini otomatis dijalankan ketika kita mengetik alamat folder saja (tanpa mengetik nama filenya).

Sebagai contoh, untuk menjalankan file **hello_world.html** yang berada di dalam folder **belajar_html**, saya harus mengetikkan alamat lengkap, yakni: **belajar_html/hello_world.html**.

Akan tetapi, untuk menjalankan file **index.html**, saya cukup mengetik alamat: **belajar_html**, dan web browser akan langsung menampilkan halaman **index.html**.

Sebelum anda mencoba hal ini, saya tambahkan bahwa ini hanya berlaku jika halaman web tersebut berada di dalam **web server**, seperti *Apache web server*. Karena pada pembahasan dalam buku ini kita tidak menggunakan web server, hal ini tidak bisa dilakukan.

Div dan Span Element

Berkaitan dengan pembahasan mengenai block dan inline element, terdapat 2 tag khusus yang sering dibahas bersamaan, yakni tag `<div>` dan tag ``. Kedua element ini tidak memiliki peranan tertentu di dalam HTML namun menjadi sangat berguna ketika dikombinasikan dengan CSS.

Tag `<div>` adalah **block level element**, sedangkan tag `` merupakan **inline level element**. Keduanya sering digunakan untuk membuat struktur halaman dengan bantuan CSS.

Baik tag `<div>` maupun tag `` bisa disebut sebagai “tag tanpa format tampilan”. Maksudnya, kedua tag ini tidak memiliki efek visual apapun.

Sebagai contoh, ketika sebuah teks ditulis ke dalam tag `<h1>`, teks tersebut akan ditampilkan dengan ukuran besar dan font tebal. Akan tetapi ketika ditulis ke dalam tag `<div>`, seolah-olah tidak ada perubahan. Kecuali terdapat sedikit spasi diatas dan dibawah teks karena tag `<div>` termasuk *block level element*.

Begini juga halnya dengan tag ``. Ketika sebuah teks berada diantara tag ``, tidak ada efek visual apapun. Beda halnya dengan tag `<i>` yang akan membuat teks tampil dalam huruf miring. Baik tag `` dan tag `<i>` termasuk ke dalam kelompok *inline level element*.

Sifat tag `<div>` dan `` yang tidak bermakna ini justru menjadikannya sebagai tag ideal untuk CSS. Karena penggunaan kedua tag ini sangat bergantung kepada CSS, maka akan menjadi jatah materi di buku **CSS Uncover** DuniaIlkom.

Dalam bab ini kita telah membahas secara mendalam tentang pengertian tag, atribut dan element dalam HTML. Kita juga telah membahas struktur dasar penulisan HTML.

Untuk bab berikutnya, saya akan membahas dengan detail tentang tag `<head>`, atau **head element**, yakni tag dan atribut apa saja yang biasanya ditempatkan di bagian `<head>` sebuah halaman HTML.

5. Head Element

Setelah membahas tentang struktur HTML dan aturan dasar penulisan HTML pada bab sebelumnya, dalam bab kali ini saya akan membahas lebih jauh tentang head element, yakni tag-tag apa saja yang umumnya ada dalam bagian `<head>` dari sebuah halaman HTML.

Tag `<head>` biasanya berisi informasi tentang halaman yang akan diproses atau pendefinisian link kepada file-file yang dibutuhkan dalam memproses halaman tersebut, seperti file CSS atau file JavaScript.

Dimanakah tag `<head>` ini?

Tag `<head>` berada pada urutan paling atas setelah penulisan tag `<html>`. Agar lebih jelas, berikut adalah kode HTML dalam file `index.html` yang telah kita buat pada bab sebelumnya:

`index.html`

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar HTML</title>
6 </head>
7 <body>
8   <h1>Belajar HTML</h1>
9   <p>Mohon <strong>jangan diganggu</strong>,
10  lagi serius belajar <em>HTML</em></p>
11 </body>
12 </html>
```

Terlihat bahwa tag `<head>` berada langsung setelah tag `<html>`. Dalam contoh tersebut, tag `<head>` berisi tag `<meta>` dan `<title>`. Selain kedua tag ini, bagian `<head>` juga bisa diisi dengan berbagai tag HTML lain. Kita akan mempelajarinya satu persatu.

5.1 Meta Element

Umumnya, element pertama yang berada di dalam tag `<head>` adalah **meta element**.

Tag `<meta>` berfungsi untuk memberi informasi mengenai halaman HTML yang saat ini sedang ditampilkan. Tag ini ditujukan bukan kepada pengunjung website, tetapi kepada web browser dan ‘*robot*’ seperti mesin pencari. Oleh karena itu efek dari tag `<meta>` tidak akan tampak bagi pengunjung.

Seluruh tag `<meta>` bersifat *opsional* dan tidak harus ada dalam setiap halaman HTML. Pengecualian untuk hal ini adalah tag `<meta charset="UTF-8">` yang jika tidak ditambahkan, validator HTML5 akan mengeluarkan error.

Di dalam tag `<meta>`, *atribut*-lah yang akan mendefinisikan fungsi dari tag `<meta>` tersebut. Atribut untuk meta tag sangat beragam, dan berbagai fungsi baru terus ditambahkan (terutama pada HTML5).

Tag meta termasuk kelompok **void element**, sehingga tidak memiliki pasangan tag penutup.

Oleh karena tag `<meta>` hadir dalam beragam fungsi, disini saya hanya membahas tag `<meta>` yang paling sering digunakan:

Meta charset

Meta charset berfungsi memberi instruksi kepada web browser tentang bagaimana cara menerjemahkan karakter-karakter yang ada di dalam halaman HTML.

Charset adalah kumpulan daftar kode-kode bit komputer dengan pasangan karakter yang harus ditampilkan. Penjelasan tentang charset ini agak bersifat teknis, namun mudah-mudahan anda bisa memahaminya.

Di dalam komputer, setiap karakter seperti huruf dan angka disimpan di dalam kumpulan bit yang terdiri dari angka 1 dan 0. Sebagai contoh, huruf "A" akan disimpan sebagai bit **1000001** atau "41" dalam *heksadesimal*. Proses penerjemahan bit "1000001" menjadi "A" merujuk kepada tabel [ASCII¹](#), atau biasa disebut dengan **charset ASCII**.

Permasalahannya, tabel charset ASCII hanya bisa menyimpan huruf dan angka latin. Untuk karakter non-latin (seperti huruf arab, china, jepang, dll), diperlukan charset khusus.

Selama perkembangan internet dan HTML, banyak bermunculan charset-charset untuk masing-masing bahasa ini. Sebagai contoh, charset **big5** digunakan untuk huruf cina, **x-euc-jp** untuk huruf jepang, dan **iso-8859-7** untuk huruf yunani. Charset yang terpisah-pisah ini menyulitkan web browser dalam menerjemahkan sebuah karakter.

Pada era XHTML, diperkenalkan charset **UTF-8** (*Unicode Transformation Format-8*) yang mendukung hampir seluruh karakter yang ada di dunia.

Dalam perkembangan selanjutnya, HTML5 juga menggunakan UTF-8 sebagai charset standar. Dengan menggunakan UTF-8, kita tidak perlu khawatir mengenai karakter atau bahasa apa yang akan digunakan. UTF-8 mendukung hingga lebih dari 10.000 karakter untuk bermacam-macam bahasa yang ada di dunia.

Di dalam HTML5, penulisan meta tag charset untuk UTF-8 adalah sebagai berikut:

```
<meta charset="UTF-8">
```

Sebagai perbandingan, di dalam HTML 4 dan XHTML penulisannya agak panjang, seperti berikut ini:

¹<http://en.wikipedia.org/wiki/ASCII>

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
```

Dari beberapa referensi², ada yang menyebutkan bahwa <meta charset="UTF-8"> sepenuhnya opsional untuk HTML5, sehingga boleh tidak ditulis. Akan tetapi, seperti yang telah anda lihat, validator HTML5 akan mengeluarkan error jika kita tidak menyertakan tag <meta> ini.

Sebagai tambahan, pada tahun 2005 terdapat sebuah bug pada Internet Explorer yang dinamakan **Google XSS**. Hal ini terjadi karena situs google tidak menuliskan karakter set sehingga bisa digunakan untuk menyisipkan kode JavaScript. Oleh karena itu, sebaiknya anda selalu menambahkan meta tag charset pada setiap halaman HTML.

Beberapa teks editor yang menyediakan template HTML, juga selalu menyertakan meta charset ini.

Meta Author, Keywords dan Description

Ketiga meta tag ini sengaja saya satukan karena fungsinya yang mirip. Ketiganya diperuntukkan untuk mempermudah ‘*robot*’ mesin pencari (seperti google) dalam mencari informasi.

Meta tag ini ditulis dengan atribut **name** untuk jenis informasi, dan atribut **content** untuk isi informasinya.

Sebagai contoh, karena situs duniaIlkom ditulis atas nama “DuniaIlkom”, maka saya membuat meta tag **author** sebagai berikut:

```
<meta name="author" content="DuniaIlkom">
```

Selanjutnya, situs duniaIlkom adalah situs tutorial bahasa pemrograman web seperti HTML, CSS, JavaScript, PHP dan MySQL. Oleh karena itu saya memutuskan untuk menggunakan kelimanya di dalam meta tag **keyword**:

```
<meta name="keywords" content="Tutorial HTML, Tutorial CSS,  
Tutorial JavaScript, Tutorial PHP, Tutorial MySQL">
```

Terakhir, untuk meta tag **description** saya menulis penjelasan mengenai situs duniaIlkom:

```
<meta name="description" content="Situs Belajar Web Programming">
```

Pada awal kemunculannya, ketiga meta tag ini digunakannya untuk mempermudah mesin pencari, namun saat ini efektifitas ketiganya sudah tidak terlalu berpengaruh. Ini terjadi karena ketiganya banyak dimanfaatkan oleh blog *spam* untuk mencurangi hasil pencarian (dengan menuliskan kata kunci sebanyak mungkin).

Walaupun demikian, tidak ada salahnya menyertakan meta tag ini pada setiap halaman HTML yang anda buat, asalkan keyword dan description yang digunakan sesuai dengan konten, dan tidak terlalu banyak hingga dianggap *spam*.

²<http://stackoverflow.com/questions/14669352/is-the-charset-meta-tag-required-with-html5>

Sekilas Tentang SEO

Bagaimana cara untuk dapat tampil di bagian teratas hasil pencarian adalah tujuan dari SEO (*Search Engine Optimization*). SEO adalah sebuah ‘ilmu’ yang membahas cara, tips dan trik agar dapat tampil teratas dalam hasil mesin pencari seperti google, yang pada akhirnya bertujuan untuk meningkatkan pengunjung ke sebuah situs.

Saat ini SEO adalah ladang subur yang ‘sangat menggiurkan’. Jika anda pemilik situs/blog dan ingin agar tampil teratas di google, maka harus mengetahui konsep SEO.

Berbagai pihak juga menyediakan jasa SEO beserta tips dan triknya. Harga yang ditawarkan mulai dari puluhan ribu hingga jutaan rupiah, dengan 1 tujuan: situs anda tampil teratas di google.

Beberapa tahun yang lalu, cukup dengan membuat ratusan kata kunci pada meta tag **keywords**, serta puluhan *backlink* (link yang menuju situs kita) akan meningkatkan ranking di Google. Akan tetapi saat ini cara tersebut tidak lagi efektif. Dan hingga saat ini pakar-pakar SEO terus mencari cara baru untuk dapat ‘mengakali’ hasil pencarian Google.

Namun 1 hal yang pasti, Google juga terus memperbarui algoritma yang menentukan apakah sebuah situs layak tampil di halaman awal pencarian atau tidak (salah satunya dengan mengurangi nilai dari meta tag **keywords**). Membuat situs dengan konten yang bermanfaat dan berguna adalah cara terbaik untuk meningkatkan SEO.

Meta Refresh

Meta tag ini cukup powerful, ia digunakan untuk membuat halaman web refresh secara otomatis setiap beberapa detik. Meta tag ini membutuhkan atribut **http-equiv** dengan nilai “refresh”, dan atribut **content** yang berisi berisi angka dalam satuan detik.

Jika anda menginginkan halaman web yang refresh setiap 1 menit, maka gunakan meta tag berikut:

```
<meta http-equiv="refresh" content="60">
```

Umumnya fitur ini ditambahkan ke dalam halaman web yang kontennya diupdate dengan sangat cepat (dalam hitungan menit atau detik) seperti situs berita. Dengan demikian, berita yang dilihat pengujung selalu berita terbaru. Beberapa web populer yang menggunakan fitur refresh adalah kaskus.co.id dan kompas.com.

Namun perlu juga diperhatikan bahwa dalam setiap siklus refresh, web browser akan memuat ulang seluruh halaman. Yang berarti akan mengurangi quota bandwidth untuk web server (tempat dimana situs berada) dan juga bandwidth pengunjung web.

Meta Tag Lainnya

Selain meta tag yang saya bahas disini, masih ada beberapa meta tag lain yang akan anda jumpai, beberapa terkait dengan penanganan CSS, JavaScript atau fitur yang dinamakan **microdata**.

Karena meta tag ini masuk kategori yang ‘*advanced*’ dan butuh pengetahuan teknis, saya hanya akan membahasnya sekilas.

Meta tag X-UA-Compatible

```
<meta http-equiv="X-UA-Compatible" content="IE=edge">
```

Meta tag **X-UA-Compatible** digunakan untuk menangani permasalahan terkait web browser Internet Explorer. IE memiliki mode **compatibility view** yang aktif ketika halaman web diakses dari komputer lokal atau diakses dari dalam intranet. Fitur ini membuat perlakuan berbeda dalam menangani halaman web (mirip *quirks mode*). Meta tag diatas akan memaksa IE untuk tidak masuk ke mode **compatibility view**.

Meta tag Viewport

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Meta tag diatas biasanya digunakan pada web dengan **desain responsive** (*responsive web design*). Secara garis besar, meta tag **viewport** ditambahkan agar desain responsive dapat tampil sempurna di dalam layar berukuran kecil (seperti tablet atau smartphone).

Meta tag Robots

```
<meta name="robots" content="index, follow">
```

Meta tag **robots** digunakan untuk memberi pesan kepada search engine bagaimana halaman web akan diproses, jika nilai atribut **content** adalah “**index, follow**”, maka mesin pencari akan mengindex halaman tersebut (dimasukkan ke dalam hasil pencarian).

Namun jika nilai atribut **content** adalah “**noindex,nofollow**”, maka isi web tidak akan dimasukkan ke hasil pencarian (mungkin halaman web bersifat rahasia dan anda tidak ingin seseorang mengaksesnya dari google).

Merangkum seluruh tag meta, berikut adalah contoh kode halaman index.html yang menggunakan tag-tag meta yang telah kita pelajari:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <meta name="author" content="DuniaIlkom">
6   <meta name="keywords" content="Tutorial HTML, Tutorial CSS,
7     Tutorial JavaScript, Tutorial PHP, Tutorial MySQL">
8   <meta name="description" content="Situs Belajar Web Programming">
9   <meta http-equiv="refresh" content="60">
```

```
10 <meta http-equiv="X-UA-Compatible" content="IE=edge">
11 <meta name="viewport" content="width=device-width, initial-scale=1">
12 <meta name="robots" content="index, follow">
13 <title>Belajar HTML</title>
14 </head>
15 <body>
16 <h1>Belajar HTML</h1>
17 <p>Mohon <strong>jangan diganggu</strong>,
18 lagi serius belajar <em>HTML</em></p>
19 </body>
20 </html>
```

Jika kita menjalankan kode diatas, tidak akan terlihat perubahan apa-apa. Karena tag `<meta>` memang tidak akan tampil di web browser, tapi hanya untuk informasi pelengkap untuk web browser.

5.2 Link Element

Tag `<link>` digunakan untuk membuat ‘*hubungan*’ antara halaman HTML dengan file lain, walaupun saat ini penggunaannya paling banyak adalah untuk menginput file CSS ke halaman HTML.

Link element memiliki beberapa *atribut*, yang paling sering digunakan adalah atribut `href`, dan atribut `rel`.

Atribut `href` (singkatan dari *hypertext references*) digunakan untuk menulis alamat lokasi file external yang dituju, sedangkan atribut `rel` (singkatan dari *relationship*) berisi jenis ‘hubungan’ dengan file tersebut.

Sebagai contoh, untuk menginput file CSS: `style.css` ke dalam halaman web, berikut adalah cara penulisannya:

```
<link href="style.css" rel="stylesheet">
```

Atribut lain untuk tag `<link>` adalah `type`. `Type` berfungsi untuk menginformasikan jenis file yang akan di-link. Jika file tersebut adalah CSS, maka nilai atribut type adalah: “`text/css`”, seperti berikut ini:

```
<link href="style.css" type="text/css" rel="stylesheet">
```

Masih berhubungan dengan CSS, atribut `media` bisa ditambahkan untuk menentukan media mana saja file CSS tersebut akan tampil, apakah `screen` (layar komputer), `print` (sewaktu dicetak), atau `all` (semua media). Berikut contohnya:

```
<link href="style.css" type="text/css" rel="stylesheet" media="all" >
```

Apabila atribut media tidak ditulis, nilai defaultnya adalah `media="all"`, yang berarti seluruh kode CSS akan diterapkan diseluruh media.

Mengenal MIME Type

Atribut type pada tag `<link>` (dan juga seluruh tag HTML yang memiliki atribut type) membutuhkan nilai dalam bentuk '**MIME type**'. **MIME type** adalah sebuah standar internet untuk membedakan tipe dan jenis dokumen.

Pada contoh penulisan `type="text/css"`, kita memberitahu web browser bahwa file yang di-link bertipe dokumen **teks** dan dalam format CSS. Jika file yang di-link adalah **JavaScript**, maka nilai MIME typenya adalah: `type="text/javascript"`.

Dalam mempelajari pemrograman web, anda akan sering berhubungan dengan MIME type. Penulisan MIME terdiri dari 2 bagian, yakni *jenis aplikasi* dan *jenis format*, dimana penulisan-nya adalah: **jenis aplikasi/jenis format**.

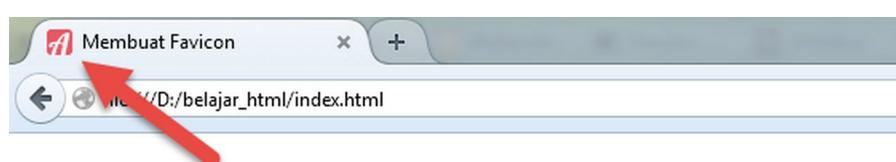
Sebagai contoh, untuk file gambar (image) dalam format GIF, maka nilai MIME typenya adalah: `image/gif`. Untuk file musik (audio) dengan format mp3, MIME typenya adalah: `audio/mpeg`.

Untuk penjelasan lebih lanjut mengenai MIME type dan jenis-jenisnya, anda bisa mempelajarinya di [wikipedia^a](http://en.wikipedia.org/wiki/Internet_media_type).

^ahttp://en.wikipedia.org/wiki/Internet_media_type

Membuat Favicon

Selain untuk menginput file CSS, tag `<link>` juga bisa digunakan untuk menginput **favicon** ke halaman HTML. **Favicon** adalah gambar kecil yang dapat anda lihat pada bagian kiri halaman web, seperti pada gambar dibawah ini.



Belajar HTML

Mohon **jangan diganggu**, lagi serius belajar **HTML**

Gambar: Tampilan favicon

Untuk menampilkan gambar ini, siapkan sebuah gambar berukuran kecil (biasanya berukuran 16×16 pixel), anda bisa membuatnya di *Photoshop* atau mencarinya di internet. File gambar ini boleh dalam format `.ico`, `.gif`, `.jpg`, atau `.png`.

Setelah itu, letakkan gambar tersebut di dalam folder **belajar_html**, dan ubah namanya menjadi **favicon.ico**, atau **favicon.jpg** (tergantung format gambar). Nama gambar ini juga tidak harus “**favicon**”, tetapi nama ini sudah menjadi standar tidak resmi dalam pembuatan web.

Langkah terakhir, silahkan buka halaman **index.html**, dan tambahkan tag berikut pada bagian **<head>**:

```
<link rel="icon" href="favicon.ico" type="image/x-icon">
```

Dalam contoh diatas, saya menggunakan format gambar **ico**. Untuk atribut **type="image/x-icon"** adalah MIME type dari file **.ico**, atribut ini boleh ditulis, boleh juga tidak (web browser akan menentukan sendiri MIME type file gambar tersebut). Jika anda menggunakan gambar **.jpg** maupun **.png**, penulisannya adalah sebagai berikut:

```
<link rel="icon" href="favicon.jpg" type="image/jpeg">
<link rel="icon" href="favicon.png" type="image/png">
```

Dengan menambahkan kode favicon, file **index.html** bisa direvisi menjadi:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <link rel="icon" href="favicon.ico" type="image/x-icon">
6   <title>Membuat Favicon</title>
7 </head>
8 <body>
9   <h1>Belajar HTML</h1>
10  <p>Mohon <strong>jangan diganggu</strong>,
11    lagi serius belajar <em>HTML</em></p>
12 </body>
13 </html>
```

Silahkan buka di web browser, akan tampil gambar kecil sebagai identitas halaman (favicon).



Saya menemukan bahwa favicon yang dijalankan secara **offline** (seperti contoh kita ini), hanya bisa tampil di web browser Mozilla Firefox. Jika dijalankan menggunakan Google Chrome maupun Opera, **favicon tidak akan tampil**³.

Favicon baru akan tampil di Google Chrome dan Opera ketika diakses via online (file berada di web hosting).

³<http://stackoverflow.com/questions/13780402/html-favicon-wont-show-on-google-chrome>

5.3 Script Element

Tag `<script>` digunakan untuk menginput kode program script ke dalam halaman HTML.

Umumnya bahasa pemrograman script yang digunakan adalah **JavaScript**, dan inilah bahasa default jika anda tidak mencantumkan jenis bahasa dalam atribut `type`.

Berikut adalah contoh penggunaan tag `<script>`:

```
<script type="text/javascript">
// kode JavaScript disini...
// kode JavaScript disini...
// kode JavaScript disini...
</script>
```

Dalam HTML5, penulisan atribut `type="text/javascript"` sepenuhnya optional, sehingga kita bisa mengabaikannya jika bahasa yang digunakan adalah JavaScript (dimana 99% kemungkinan akan menggunakan JavaScript), seperti contoh berikut:

```
<script>
// kode JavaScript disini...
// kode JavaScript disini...
// kode JavaScript disini...
</script>
```

Selain menginput langsung kode script ke dalam halaman, tag `<script>` juga digunakan untuk menginput file JavaScript external (yang dibuat terpisah) ke dalam halaman HTML. Untuk ini kita menambahkan atribut `src`. Nilai dari atribut `src` adalah lokasi tempat file script berada. Berikut contoh penulisannya:

```
<script src="javascript.js"></script>
```

Kode diatas akan menginput file `javascript.js` yang berada di dalam satu folder dengan file saat ini ke dalam halaman HTML.

Walaupun saya membahas tag `<script>` di dalam *head element*, penggunaan tag ini tidak harus di bagian `<head>` saja. Tag `<script>` bisa ditempatkan di bagian manapun sepanjang halaman HTML.

5.4 Style Element

Tag `<style>` digunakan untuk menginput kode *style* ke dalam halaman HTML. Kode *style* yang umumnya digunakan (dan masih satu-satunya) adalah CSS. CSS juga menjadi *default style* jika kita tidak menulis atribut `type="text/css"`.

Berikut adalah contoh penggunaan tag `<style>`:

```
<style type="text/css">  
// kode CSS disini  
// kode CSS disini  
</style>
```

Sama seperti tag `<script>`, atribut `type` juga bersifat *opsional* di dalam HTML5. Anda boleh tidak menuliskan atribut ini, seperti contoh berikut:

```
<style>  
// kode CSS disini  
// kode CSS disini  
</style>
```

Tag `<style>` juga mendukung atribut `media`. Atribut ini berfungsi untuk membuat *style* CSS yang hanya berjalan jika media yang ditulis sesuai. Apakah itu `screen` (layar komputer), `print` (sewaktu dicetak), atau `all` (semua media). Berikut contohnya:

```
<style media="all">  
// kode CSS disini  
// kode CSS disini  
</style>
```

Jika atribut `media` tidak ditulis, nilai defaultnya adalah “`all`”, yang berarti seluruh kode CSS akan diterapkan diseluruh media.

Sama seperti tag `<script>`, tag `<style>` juga bisa ditempatkan di bagian mana saja di dalam halaman HTML (tidak harus di bagian head).

5.5 Title Element

Tag `<title>` mungkin menjadi tag paling penting di bagian `<head>`. Tag ini berfungsi untuk menampilkan judul halaman di bagian ‘bingkai atas’ web browser.

Umumnya mesin pencari menggunakan nilai tag ini untuk menampilkan judul hasil pencarian. Sehingga membuat isi tag `<title>` yang tepat akan meningkatkan peluang tampil di Google.

Pada hasil pencarian Google, hanya 50-60 karakter pertama saja yang akan ditampilkan. Oleh karena itu sebaiknya tidak menggunakan judul yang terlalu panjang.

Jika terdapat pengunjung web yang membagi (*sharing*) ke situs media sosial seperti Facebook, judul artikel dari sharing tersebut diambil dari nilai tag `<title>` ini.

Berikut contoh penulisan tag `<title>`:

```
<title>Belajar HTML untuk pemula | DuniaIlkom</title>
```

Selain membuat judul halaman, saya juga menambahkan nama situs dibagian akhir dengan menggunakan pembatas “|”. Cara penulisan seperti ini umum dilakukan, dengan catatan judul halaman tidak melebihi 50-60 karakter.

Berikut contoh kode HTML lengkapnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar HTML untuk pemula | DuniaIlkom</title>
6 </head>
7 <body>
8   <h1>Belajar HTML</h1>
9   <p>Mohon <strong>jangan diganggu</strong>,
10  lagi serius belajar <em>HTML</em></p>
11 </body>
12 </html>
```



Belajar HTML

Mohon **jangan diganggu**, lagi serius belajar **HTML**

Gambar: Tampilan title dari sebuah halaman web

5.6 HTML5Shiv

Walaupun buku ini saya rancang untuk pemula, tetapi tidak ada salahnya saya masuk ke dalam topik yang sedikit '*advanced*'. Karena saya ingin anda mengetahui tentang perkembangan HTML5 dan 'trik' yang digunakan untuk mengatasinya.



Jika anda tidak terlalu memahami maksud dari penjelasan ini, silahkan lanjut ke bab selanjutnya. Pembahasan mengenai **HTML5Shiv** bukanlah hal wajib.

Seperti yang telah anda baca pada bab 2 tentang **Sejarah HTML**, HTML5 baru muncul beberapa tahun yang lalu. Sebagian besar kode-kode yang ada di HTML5 masih berasal dari versi HTML terdahulu, namun HTML5 juga memperkenalkan beberapa tag baru seperti `<header>`, `<article>`, `<aside>`, dll (yang nantinya akan kita bahas lebih dalam).

Masalahnya, web browser "tua" seperti Internet Explorer 6, 7 dan 8, serta beberapa versi Mozilla Firefox awal, '*tidak mengerti*' tentang tag-tag baru yang ada di dalam HTML5.

Mekanisme standar ketika sebuah web browser tidak paham dengan sebuah tag, adalah dengan melewatkannya (secara teknis, tag tersebut akan dianggap sebagai "**empty element**"). Hal ini bisa

mengubah tampilan website menjadi kacau, terutama jika tag tersebut di-*style* menggunakan CSS.

Beberapa web developer tidak ambil pusing atas masalah ini. Mereka beranggapan bahwa pengguna yang masih menggunakan web browser ‘tua’ seperti Internet Explorer 6 dan 7, tidak terlalu signifikan (mungkin hanya berkisar 1% atau 2% saja dari total pengunjung).

Tren kedepannya juga seperti ini. Web browser jadul akan semakin berkurang, dan banyak pengguna yang beralih ke web browser modern seperti Google Chrome, Mozilla Firefox, atau web browser pengganti IE: [Microsoft Edge](#)⁴.

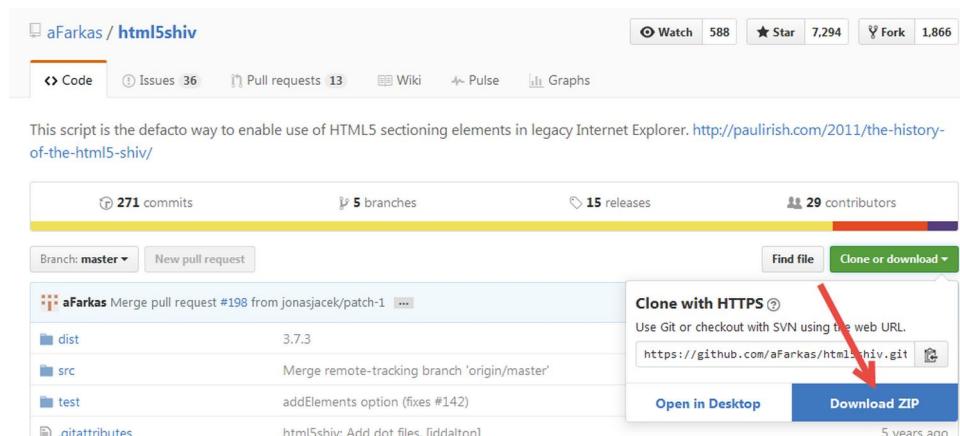
Jika anda berminat mencari cara agar HTML5 tetap bisa tampil di web browser jadul, salah satu solusinya adalah dengan meminta bantuan **JavaScript**. Menggunakan **JavaScript**, kita bisa ‘mengajari’ web browser lawas agar mengerti HTML5. Anda tidak perlu membuat kode program ini, karena terdapat beberapa *script* siap pakai yang tinggal diinput ke dalam HTML. Salah satunya adalah **HTML5Shiv**.

HTML5Shiv adalah kumpulan kode program **JavaScript** yang bertujuan agar web browser tua seperti Internet Explorer 6-9, Safari 4.x (dan iPhone 3.x), serta Firefox 3.x bisa mengerti tag-tag standar yang ada di HTML5.

Untuk menggunakan **HTML5Shiv**, terdapat 2 solusi, yakni mendownload file Javascript tersebut dan mengaksesnya dari komputer lokal, atau menggunakan file yang telah tersedia di **CDN** (*Content Delivery Network*). Kita akan bahas kedua cara ini.

Cara Mengakses **HTML5Shiv** dari Komputer Lokal

Untuk mengakses **HTML5Shiv** secara lokal, pertama kali kita harus mendownload file **JavaScript** tersebut yang beralamat di <https://github.com/aFarkas/html5shiv>⁵. Cari tombol **Download ZIP** pada sisi kanan bawah, dan web browser akan mendownload file yang kita butuhkan (berukuran sekitar **136kb**).



Gambar: Cara mendownload **HTML5shiv** dari GitHub

Silahkan extract file ZIP tersebut (**html5shiv-master.zip**), kemudian buka folder **dist**.

⁴<https://www.microsoft.com/en-us/windows/microsoft-edge>

⁵<https://github.com/aFarkas/html5shiv>

Name	Date modified	Type	Size
html5shiv.js	14/03/2015 5:30	JScript Script File	11 KB
html5shiv.min.js	14/03/2015 5:30	JScript Script File	3 KB
html5shiv-printshiv.js	14/03/2015 5:30	JScript Script File	16 KB
html5shiv-printshiv.min.js	14/03/2015 5:30	JScript Script File	5 KB

Gambar: Isi folder dist dari html5shiv-master

Dalam folder ini terdapat 4 buah file JavaScript:

- File **html5shiv.js** adalah file HTML5Shiv normal dengan berbagai komentar di dalam file tersebut agar mudah dipelajari.
- File **html5shiv.min.js** adalah versi *minified* dari html5shiv.js, dimana kode-kode yang ada telah dikompresi agar berukuran kecil (fungsinya tetap sama, tetapi kita tidak akan bisa mempelajari kode-kode yang ada).
- File **html5shiv-printshiv.js** adalah file HTML5Shiv dengan tambahan kode untuk memperbaiki hasil print pada web browser IE.
- File **html5shiv-printshiv.min.js** adalah versi minified dari html5shiv-printshiv.js.

Kali ini saya akan menggunakan file **html5shiv.js**. Silahkan copy file tersebut ke dalam folder **belajar_html**.

Untuk dapat mengakses file ini, kita tinggal memanggilnya dengan menggunakan tag `<script>` seperti berikut ini:

```
<script src="html5shiv.js"></script>
```

Silahkan letakkan tag tersebut pada bagian `<head>` dari halaman HTML. Sehingga halaman index.html akan menjadi sebagai berikut:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Menggunakan HTML5shiv</title>
6   <script src="html5shiv.js"></script>
7 </head>
8 <body>
9   <h1>Belajar HTML</h1>
10  <p>Mohon <strong>jangan diganggu</strong>,
11    lagi serius belajar <em>HTML</em></p>
12 </body>
13 </html>

```

Mengakses HTML5Shiv dari CDN

Selain mengakses file `html5shiv.js` secara manual dari komputer lokal, kita juga bisa menggunakan file `html5shiv.js` yang telah tersedia di *CDN*. *CDN* (singkatan dari *Content Delivery Network*) adalah sejenis “harddisk” global dimana kita bisa mengakses file-file yang ada di *CDN* langsung dari kode program.

Terdapat banyak penyedia layanan *CDN*, yang terkenal seperti *CDNjs*, *BootstrapCDN*, *CloudFlare*, *Maxcdn*, *Google CDN*, dll.

Kali ini saya akan menggunakan *CDNjs*, dimana file `html5shiv.js` dapat diakses dari alamat `https://cdnjs.cloudflare.com/ajax/libs/html5shiv/3.7.3/html5shiv.js`. Untuk menggunakannya di kode HTML kita, silahkan input kode berikut di bagian `<head>`:

```
<script type="text/javascript"
    src="https://cdnjs.cloudflare.com/ajax/libs/html5shiv/3.7.3/html5shiv.js">
</script>
```

Sehingga file `index.html` akan menjadi:

```
index.html


---


1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="UTF-8">
5     <title>Belajar Menggunakan CDN HTML5shiv</title>
6     <script type="text/javascript"
7         src="https://cdnjs.cloudflare.com/ajax/libs/html5shiv/3.7.3/html5shiv.js">
8     </script>
9 </head>
10 <body>
11     <h1>Belajar HTML</h1>
12     <p>Mohon <strong>jangan diganggu</strong>,
13         lagi serius belajar <em>HTML</em></p>
14 </body>
15 </html>
```

Salah satu keuntungan menggunakan *CDN* adalah kita bisa memanfaatkan fitur **cache** di web browser.

Karena *CDN* dapat digunakan oleh seluruh website di internet, besar kemungkinan ada website lain yang juga menggunakan *HTML5Shiv* pada *CDNjs*.

Pada saat user mengunjungi sebuah situs yang menggunakan *HTML5Shiv* dari *CDNjs*, web browser akan menyimpan file `html5.js` tersebut di cache web browser (penyimpanan sementara). Jika ada situs lain yang juga menggunakan file `html5.js` dari *CDNjs*, web browser tinggal menggunakan file yang ada di cache, tanpa perlu mendownload ulang file tersebut.

HTML5Shiv hanya di IE

Apabila kita menggunakan salah satu cara diatas untuk menginput **HTML5Shiv**, file **html5shiv.js** akan tetap berjalan meskipun pengunjung menggunakan web browser modern yang telah mendukung HTML5.

Agar lebih efisien, kita bisa menggunakan perintah tambahan agar **HTML5Shiv** hanya dijalankan ketika web diakses dari Internet Explorer saja (lebih spesifik lagi, IE 9 kebawah, karena IE 9 keatas sudah mendukung HTML5). Berikut adalah kode yang dibutuhkan:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Menggunakan CDN HTML5shiv IE</title>
6   <!--[if lt IE 9]>
7     <script type="text/javascript"
8       src="https://cdnjs.cloudflare.com/ajax/libs/html5shiv/3.7.3/html5shiv.js">
9     </script>
10    <![endif]-->
11 </head>
12 <body>
13   <h1>Belajar HTML</h1>
14   <p>Mohon <strong>jangan diganggu</strong>,
15     lagi serius belajar <em>HTML</em></p>
16 </body>
17 </html>
```

Tag `<!--[if lt IE 9]>` adalah tag khusus yang hanya bisa dibaca oleh IE. Pada web browser selain IE, kode diatas diproses sebagai komentar (perhatikan awal tag `<!--` yang merupakan cara penulisan komentar di HTML).

Kode `if lt IE 9` adakah singkatan dari “**if less than IE 9**”, yang berarti: *jalankan kode berikut hanya pada IE sebelum 9*, sehingga kode diatas akan dijalankan pada web browser IE 6, 7, dan 8. Tag `<![endif]-->` digunakan sebagai tag penutup.

5.7 Penutup: Head Element

Dalam bab ini kita telah membahas beberapa hal yang berkaitan dengan element atau tag HTML yang sering terdapat di dalam bagian `<head>` sebuah halaman web.

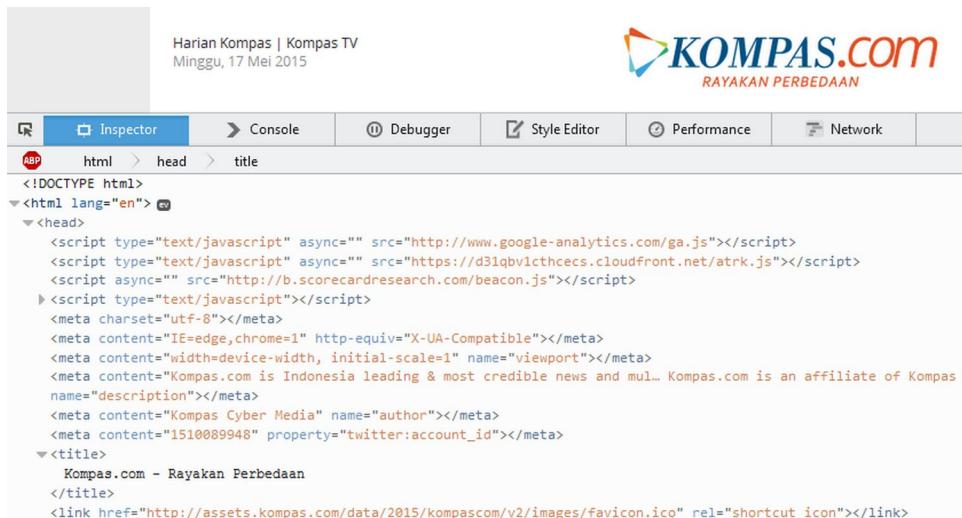
Sebagai penutup, saya akan menampilkan file **index.html** dengan seluruh tag-tag yang telah kita pelajari sejauh ini. Berikut adalah kode HTML lengkapnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <meta name="author" content="DuniaIlkom">
6   <meta name="keywords" content="Tutorial HTML, Tutorial CSS,
7     Tutorial JavaScript, Tutorial PHP, Tutorial MySQL">
8   <meta name="description" content="Situs Belajar Web Programming">
9   <meta http-equiv="refresh" content="60">
10  <meta http-equiv="X-UA-Compatible" content="IE=edge">
11  <meta name="viewport" content="width=device-width, initial-scale=1">
12  <meta name="robots" content="index, follow">
13  <link href="style.css" type="text/css" rel="stylesheet" media="all">
14  <link rel="icon" href="favicon.ico" type="image/x-icon">
15  <!--[if lt IE 9]>
16    <script type="text/javascript"
17      src="https://cdnjs.cloudflare.com/ajax/libs/html5shiv/3.7.3/html5shiv.js">
18    </script>
19  <![endif]-->
20  <script src="javascript.js"></script>
21  <title>Belajar Head Element HTML</title>
22  <style media="all">
23    /* kode CSS disini
24      kode CSS disini */
25  </style>
26 </head>
27 <body>
28  <h1>Belajar Head Element HTML</h1>
29  <p>Mohon <strong>jangan diganggu</strong>,
30    lagi serius belajar <em>HTML</em></p>
31 </body>
32 </html>
```

Sampai disini kita telah membahas sebagian besar tag beserta atribut yang sering muncul di bagian `<head>` dari sebuah file HTML.

Walaupun anda merasa informasi yang saya jelaskan disini terasa cukup panjang, tetapi ini masih sebagian kecil dari apa yang bisa diletakkan pada tag `<head>`. Sebagai perbandingan dan studi kasus, anda bisa buka website populer seperti kaskus.co.id, kompas.com atau detik.com, dan lihat source codenya (terutama di bagian `<head>`).



Gambar: HTML bagian <head> dari situs kompas.com

Beberapa hal yang mungkin akan anda temui adalah atribut khusus seperti `property="twitter:account_id"`, atau `property="fb:app_id"`. Ini adalah *metatag* khusus yang digunakan untuk sosial media seperti *Twitter* dan *Facebook*.

Beberapa atribut lain yang mungkin anda temui seperti `itemprop="title"` atau `property="og:site_name"`. Atribut-atribut ini dikenal dengan sebutan **microdata**. Microdata biasanya digunakan untuk memandu *search engine* untuk menterjemahkan beberapa informasi.

Sebagai buku pengantar HTML untuk pemula, saya rasa pembahasan mengenai microdata tidak terlalu ‘urgent’, tapi jika anda berminat mempelajarinya, bisa mengunjungi [wikipedia⁶](http://en.wikipedia.org/wiki/Microdata_%28HTML%29) atau [schema.org⁷](http://schema.org/).

Kita tinggalkan bagian <head> dan masuk ke bagian <body> dari halaman HTML. Dalam pembahasan pada bab selanjutnya, kita akan membahas tentang **Text Formatting Element**, yakni tag-tag yang digunakan untuk menformat teks atau paragraf di dalam HTML.

⁶http://en.wikipedia.org/wiki/Microdata_%28HTML%29

⁷schema.org

6. Text Formatting Element

Dalam bab kali ini kita akan membahas tag, atribut dan element yang berkaitan dengan memformat teks, seperti cara membuat paragraf, link, huruf miring, huruf tebal, dll. Semua tag-tag ini ditempatkan di dalam bagian <body> dari halaman web.

Perlu menjadi catatan, ketika membahas tag atau atribut yang memiliki efek ‘tampilan’, biasanya tag atau atribut tersebut telah dinyatakan **deprecated** (usang), dan kita disarankan untuk beralih menggunakan CSS.

Agar pembahasan materi lebih komplit, saya tetap membahas tag dan atribut ‘usang’ ini namun juga menyertakan contohnya dengan CSS. CSS adalah pembahasan yang sangat luas, dan umumnya dibahas dalam buku tersendiri. Walaupun demikian, di akhir buku saya menyertakan panduan singkat memahami CSS. Ini saya buat agar anda bisa mengikuti pembahasan materi yang ‘sedikit’ melibatkan CSS.



Agar menghemat tempat, beberapa contoh kode HTML saya buat hanya sebagian saja. Untuk mencobanya, silahkan langsung tempatkan di bagian <body> dari halaman **index.html** yang kita gunakan dalam bab sebelum ini.

6.1 Paragraf Element

Tag <p> digunakan untuk membuat paragraf, dan besar kemungkinan tag inilah yang paling banyak anda gunakan ketika mengisi konten untuk sebuah website.

Untuk membuat paragraf, kita tinggal mengisi teks diantara tag pembuka <p> dan tag penutup </p>, seperti contoh berikut:

```
<p>Ini adalah paragraf pertama</p>
<p>Ini adalah paragraf kedua</p>
```

Tag <p> merupakan **block level element**, sehingga akan tampil terpisah dalam baris baru. Web browser juga akan menambahkan beberapa spasi sebelum dan sesudah paragraf. Spasi ini (atau lebih tepatnya: *margin*) nantinya bisa dihilangkan atau diubah menggunakan CSS.

Memformat Rata Tepi Paragraf (atribut align)

Salah satu hal yang ingin anda lakukan di dalam paragraf adalah mengatur rata tepi (*align*) sebuah teks. Apakah teks itu akan ditampilkan dengan rata kanan, kiri, tengah, atau keduanya (*justify*). Untuk mendapatkan hasil ini, kita bisa menggunakan atribut **align**. Nilai yang bisa diinput adalah **left**, **right**, **center**, atau **justify**.

Berikut contoh penggunaan atribut align di dalam tag <p>:

```
<p align="center">  
    Lorem ipsum dolor sit amet, consectetur adipiscing elit.  
    Nam dictum sapien nec elit volutpat pellentesque.  
    Vestibulum tristique justo ut eros iaculis, sit amet tempor luctus.  
    Quisque convallis sapien sit amet sem efficitur, at efficitur ornare.  
</p>
```

Namun sebelum anda menambahkan atribut ini ke dalam HTML, atribut **align** sudah dinyatakan **deprecated** oleh standar HTML5.

Mengenal Pengertian Deprecated

Deprecated adalah istilah bahasa inggris yang berarti ‘*usang*’ atau ‘*tercela*’. Dalam bahasa pemrograman komputer, istilah *deprecated* berarti sebuah hal yang tidak didukung lagi, sudah usang dan sebaiknya tidak dipakai.

Sebuah tag atau atribut HTML jika dinyatakan sebagai *deprecated*, berarti tag atau atribut tersebut sangat disarankan untuk tidak digunakan. Kedepannya, tag atau atribut deprecated mungkin tidak akan di support oleh web browser.

Dalam HTML, tag atau atribut *deprecated* biasanya berasal dari versi HTML terdahulu dan berkaitan dengan efek tampilan. Sebagaimana yang telah kita pelajari dalam bab pertama, HTML dirancang untuk membuat struktur, sedangkan masalah tampilan diserahkan kepada CSS.

Selain deprecated, terdapat juga istilah **obsolete**. Kedua istilah ini bermakna mirip, yakni menandakan sesuatu yang sebaiknya tidak digunakan lagi. Pada spesifikasi HTML5, istilah yang sering dipakai adalah **obsolete**, namun dalam bahasa pemrograman secara umum, istilah deprecated lebih sering digunakan.

Dalam contoh kita kali ini atribut align sudah berstatus *deprecated*, oleh karena itu kita disarankan untuk menggunakan property CSS: `text-align` untuk mendapatkan efek yang sama.

Mengatur Align Paragraf Dengan CSS

Hampir semua atribut HTML yang berstatus *deprecated* memiliki pasangan kode CSS untuk menggantikannya, tidak terkecuali atribut **align**. Efek rata teks yang sama bisa di dapat dengan menggunakan *property text-align* dari CSS. Nilai dari *text-align* ini sama seperti atribut align, yakni salah satu dari: **left**, **right**, **center**, atau **justify**.

Berikut contoh penggunaan property *text-align* untuk tag `<p>`:

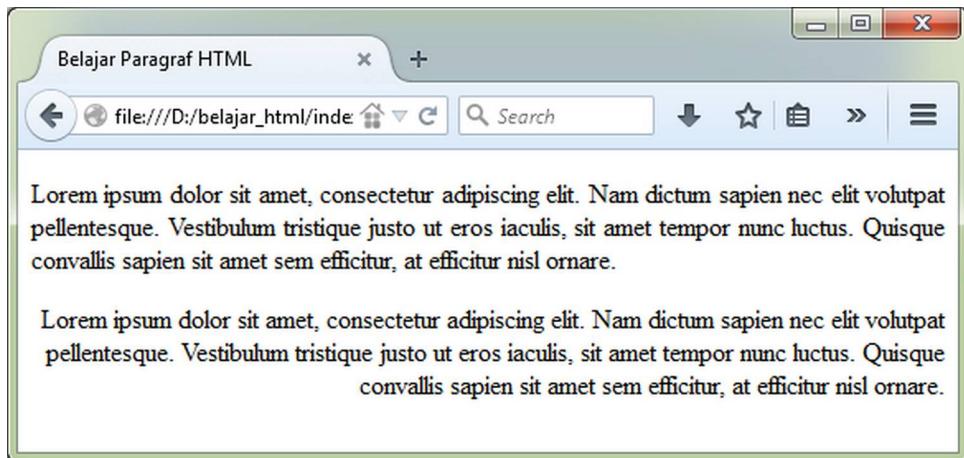
```
<p style="text-align: justify">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    Nam dictum sapien nec elit volutpat pellentesque.
    Vestibulum tristique justo ut eros iaculis, sit amet tempor luctus.
    Quisque convallis sapien sit amet sem efficitur, at efficitur ornare.
</p>

<p style="text-align: right">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    Nam dictum sapien nec elit volutpat pellentesque.
    Vestibulum tristique justo ut eros iaculis, sit amet tempor luctus.
    Quisque convallis sapien sit amet sem efficitur, at efficitur ornare.
</p>
```

Berikut kode lengkapnya pada halaman **index.html**:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="UTF-8">
5     <title>Belajar Paragraf HTML</title>
6 </head>
7 <body>
8     <p style="text-align: justify">
9         Lorem ipsum dolor sit amet, consectetur adipiscing elit.
10        Nam dictum sapien nec elit volutpat pellentesque.
11        Vestibulum tristique justo ut eros iaculis, sit amet tempor luctus.
12        Quisque convallis sapien sit amet sem efficitur, at efficitur ornare.
13     </p>
14     <p style="text-align: right">
15         Lorem ipsum dolor sit amet, consectetur adipiscing elit.
16         Nam dictum sapien nec elit volutpat pellentesque.
17         Vestibulum tristique justo ut eros iaculis, sit amet tempor luctus.
18         Quisque convallis sapien sit amet sem efficitur, at efficitur ornare.
19     </p>
20 </body>
21 </html>
```



Gambar: Cara penggunaan property CSS text align pada tag <p>

Seperti layaknya sebuah paragraf, di dalam tag <p> kita juga bisa menggunakan berbagai **inline level element** sebagai isi dari paragraf, seperti membuat huruf miring, huruf tebal, atau link.

Berkenalan dengan Lorem Ipsum

Lorem Ipsum adalah salah satu hal unik yang akan sering anda jumpai pada contoh-contoh kode program atau contoh tutorial (terutama ketika membahas web desain).

Lorem ipsum merupakan *dummy text* atau *placeholder text* yang digunakan sebagai pengganti teks. Tujuannya, agar tampilan kita ‘berisi sesuatu’ (dimana isi dari teks tersebut tidaklah penting, karena yang ingin dilihat adalah efek tampilannya).

Lorem ipsum telah digunakan sejak tahun 1500an di industri percetakan. Teks dengan bahasa latin ini berasal dari sebuah karya sastra pada tahun 45 sebelum masehi. Jika anda tertarik tentang asal-usul Lorem Ipsum ini, bisa membacanya di [wikipedia^a](#).

Saya sendiri lebih menyukai membuat contoh menggunakan **lorem ipsum** agar perhatian kita tidak terpecah dengan ‘isi’ teks, dan lebih fokus kepada pembahasan HTML.

^ahttp://en.wikipedia.org/wiki/Lorem_ipsum

6.2 Anchor Element (a)

Tag <a> (disebut juga dengan *anchor element*) merupakan salah satu tag terpenting dalam HTML. Tag ini digunakan untuk membuat **link** ke halaman lain (atau bisa juga untuk membuat link kepada file gambar, file pdf, file mp3, dll).

Untuk membuat link, setidaknya perlu 2 hal, yakni teks yang berfungsi sebagai teks link (teks yang bisa ‘di-klik’) dan alamat tujuan link (**URL**).

Untuk membuat teks yang bisa di-klik, kita tinggal membuat teks diantara tag pembuka <a> dengan tag penutup , seperti contoh berikut ini:

```
<a> Halaman Login</a>
```

Sedangkan untuk membuat lokasi yang akan dituju ketika teks tersebut di klik, kita membutuhkan atribut href.

Tag <a> termasuk ke dalam *inline level element*, sehingga ia akan ditampilkan mengikuti baris teks sebelumnya (tidak pindah ke baris baru).

Atribut href

Untuk menginput lokasi tujuan yang ingin di-link kan, tag <a> membutuhkan atribut href (*hypertext references*). Nilai dari atribut href berupa alamat dari halaman yang ingin dituju. Alamat ini bisa berbentuk **alamat absolut** atau **alamat relatif** (kita akan bahas keduanya sesaat lagi).

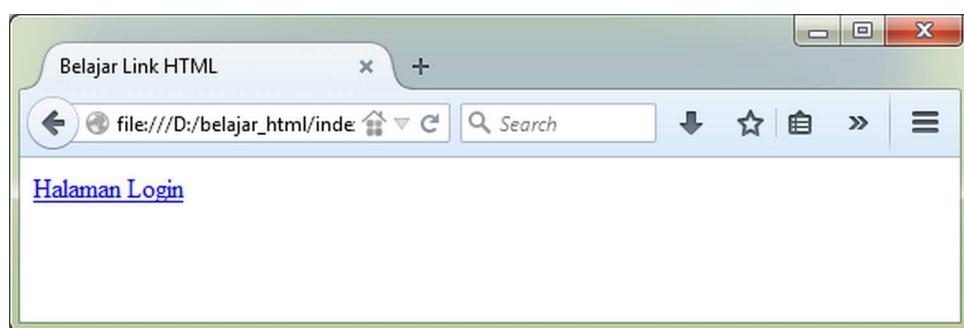
Berikut contoh penulisan tag <a> dengan penambahan atribut href:

```
<a href="http://www.duniailkom.com/login.html">Halaman Login</a>
```

Jika anda menginput kode diatas ke bagian <body> halaman index.html, berikut adalah kode lengkapnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Link HTML</title>
6 </head>
7 <body>
8   <a href="http://www.duniailkom.com/login.html">Halaman Login</a>
9 </body>
10 </html>
```



Gambar: Cara penggunaan atribut href pada tag <a>

Teks **Halaman Login** berfungsi sebagai *anchor teks*, yakni teks yang bisa di-klik. Secara default, teks ini berwarna biru dengan efek garis bawah (*underline*). Kita bisa mengubah efek ini dengan menggunakan CSS.

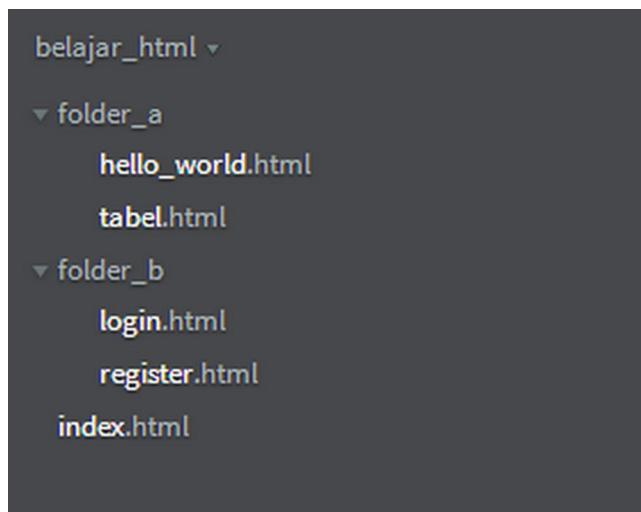
Mengenal Alamat Absolut dan Alamat Relatif

Secara umum, alamat lokasi file di dalam HTML dapat dikelompokkan menjadi dua jenis, yakni alamat absolut dan alamat relatif.

Alamat absolut adalah alamat yang mencantumkan URL secara lengkap, yakni terdiri dari nama protocol (bagian `http://`), nama domain (seperti: `www.duniaIlkom.com`), dan nama file (seperti: `login.html`). Contoh alamat absolut secara lengkap adalah: `http://www.duniaIlkom.com/login.html`.

Alamat relatif adalah alamat yang ‘*relatif*’ kepada file tujuan berdasarkan struktur folder. Dengan membuat alamat relatif, kita tidak perlu membuat alamat URL lengkap seperti alamat absolut. Namun alamat relatif ini perlu pembahasan khusus.

Agar lebih mudah memahami cara penulisan alamat relatif, saya akan menggunakan contoh struktur folder seperti pada gambar dibawah ini.



Gambar: Contoh struktur file

Dalam struktur diatas, saya memiliki folder `belajar_html` yang di dalamnya terdapat 2 folder: `folder_a` dan `folder_b` serta sebuah file: `index.html`. Dalam `folder_a` terdapat 2 file: `hello_world.html` dan `tabel.html`. Di dalam `folder_b` juga terdapat 2 file: `login.html` dan `register.html`.

Misalkan saat ini saya sedang mengetik kode pada file `index.html`. Untuk membuat link ke halaman `tabel.html` yang berada pada `folder_a`, saya tinggal menulis lokasi dari file `tabel.html` relatif kepada file `index.html`. Berikut cara penulisan link-nya:

```
<a href="folder_a/tabel.html">Halaman Tabel</a>
```

Apabila file yang saya tuju adalah `login.html` dalam `folder_b`, maka penulisannya menjadi:

```
<a href="folder_b/login.html">Halaman Login</a>
```

Sebagai contoh lain, sekarang saya akan menggunakan file `hello_world.html`. Untuk membuat link dari halaman `hello_world.html` ke halaman `tabel.html`, saya cukup menuliskan alamat: `tabel.html` (karena kedua file ini berada di dalam folder yang sama). Berikut cara penulisan link-nya:

```
<a href="tabel.html">Halaman Tabel</a>
```

Namun jika saya ingin membuat link ke file *index.html*, maka saya harus ‘naik’ 1 folder ke atas. Untuk naik 1 folder, kita menulisnya dengan karakter titik dua kali diikuti dengan garis miring: ‘..<’. Sehingga penulisan linknya menjadi seperti berikut:

```
<a href=". ./index.html">Halaman Index</a>
```

Masih menggunakan file *hello_world.html*, jika saya ingin membuat link ke file *register.html*, maka saya harus naik 1 folder, pindah ke *folder_b*, dan menuliskan nama file *register.html*. Berikut cara penulisan link-nya:

```
<a href=". ./folder_b/register.html">Halaman Register</a>
```

Aturan penulisan alamat ini sangat penting untuk dipahami, karena penggunaannya tidak hanya untuk tag saja, tetapi juga untuk mengakses file external lain seperti gambar, file CSS, atau file JavaScript.

Anda mungkin akan menemukan struktur folder yang cukup rumit, sehingga harus naik beberapa folder, kemudian masuk ke dalam beberapa folder lagi untuk menemukan file yang dituju, seperti contoh berikut:

```
<a href=". ./. ./folder_a/folder_b/terpendam.html">Halaman Terpendam</a>
```

Dalam contoh diatas, kita naik 3 folder (dari file saat ini), kemudian masuk ke *folder_a*, ke *folder_b*, dan akhirnya menemukan file *terpendam.html*.



Salah satu keuntungan menggunakan alamat relatif adalah kita tidak perlu mengubah link jika pindah domain situs. Sebagai contoh, apabila saya menggunakan alamat relatif untuk seluruh link internal di *duniaIlkom.com*, maka saya tidak perlu mengubah alamat link yang ada jika duniaIlkom pindah alamat ke *duniailmukomputer.com*.

Link ke Bagian Lain pada Halaman yang Sama

Selain berfungsi untuk membuat link ke halaman lain, tag juga bisa digunakan untuk membuat link ke bagian lain pada halaman yang sama. Hal ini lazim dilakukan jika halaman yang anda tulis cukup panjang, sehingga memerlukan ‘daftar isi’ pada bagian awal. Jika daftar isi ini di klik, maka layar web browser akan pindah ke bagian tersebut.

Untuk membuat ini, kita perlu melakukan 2 hal: menandai lokasi yang dituju, dan membuat link menuju lokasi tersebut.

Menandai sebuah lokasi yang akan menjadi tujuan link, bisa dilakukan dengan menggunakan atribut *id*. Atribut *id* merupakan salah satu *atribut global*. *Atribut global* atau *global attribute* adalah atribut yang bisa digunakan oleh seluruh tag HTML.

Nilai dari atribut *id* biasanya berupa sebuah kata atau kumpulan karakter (kita juga bisa menggunakan angka, garis penghubung, atau underscore). Sebagai contoh, berikut adalah penulisan atribut *id* dengan nilai *bab1* pada sebuah paragraf:

```
<p id="bab1">Ini adalah paragraf pembuka di dalam bab 1</p>
```



Selain sebagai penanda link, atribut `id` juga berfungsi untuk penanda tag ketika menggunakan CSS atau JavaScript.

Setelah lokasi link selesai dibuat, kita tinggal ‘mengaitkan’ nilai atribut `id` tersebut dengan atribut `href`. Caranya, dengan menggunakan tanda pagar (#) dan diikuti dengan nilai atribut `id` yang dituju.

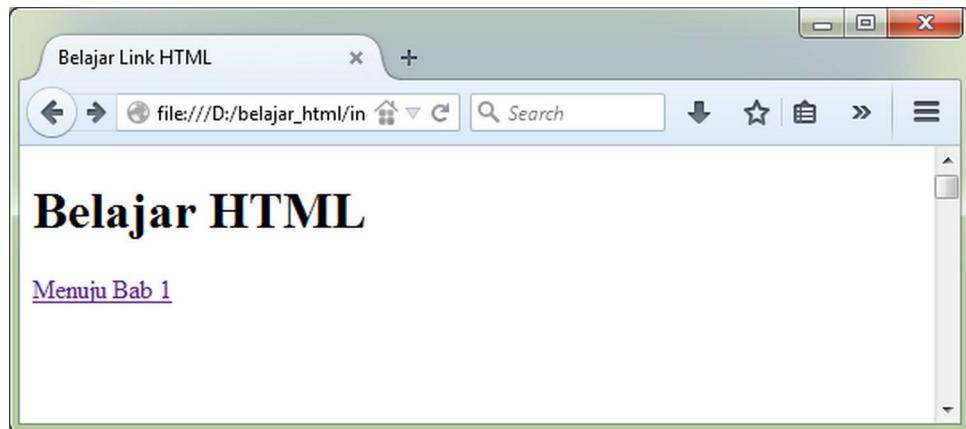
Untuk membuat link yang akan menuju tag dengan `id="bab1"`, saya tinggal menulis atribut `href="#bab1"` pada tag `<a>`. Notasi penulisan atribut `href` ini disebut juga dengan *fragment identifier*. Berikut adalah penulisan linknya:

```
<a href="#bab1">Menuju Bab 1</a>
```

Sebagai file latihan, saya akan memodifikasi file `index.html` menjadi seperti berikut ini:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Link HTML</title>
6 </head>
7 <body>
8   <h1>Belajar HTML</h1>
9   <a href="#bab1">Menuju Bab 1</a>
10  <br><br><br><br><br><br><br><br><br><br><br><br><br>
11  <br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>
12  <br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>
13  <br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>
14  <p id="bab1">Ini adalah paragraf pembuka di dalam bab 1</p>
15 </body>
16 </html>
```



Gambar: Contoh link ke halaman yang sama, dengan atribut id

Pada contoh diatas, saya menggunakan banyak tag
 agar teks paragraf tidak terlihat pada jendela web browser (seolah-olah berada ‘jauh’ dibawah), sehingga ketika link “*Menuju Bab 1*” di klik, web browser akan pindah ke bagian tersebut.

Anda juga bisa mengkombinasikan nilai atribut href menjadi seperti berikut ini:

```
<a href="http://www.duniaikom.com/tutorial.html#bab2">  
    Tutorial bab 2  
</a>
```

Ketika link diatas di-klik, web browser akan menampilkan halaman **tutorial.html** pada situs duniaikom.com, dan langsung menuju **bab 2**.

Atribut Target

Atribut **target** pada tag <a> digunakan untuk menentukan pada jendela mana halaman web akan ditampilkan. Nilai yang bisa digunakan untuk atribut ini adalah **_self**, **_blank**, **_parent**, **_top**, atau **framename**.

Apabila kita menggunakan atribut **target="_self"**, maka web browser akan menampilkan halaman link di dalam jendela saat ini. Berikut contoh penggunaannya:

```
<a href="http://www.duniaikom.com/tutorial.html" target="_self">  
    Halaman Tutorial  
</a>
```

Atribut **target="_self"** jarang ditulis, karena jika tidak ditulis pun link akan tetap tampil di halaman saat ini.

Atribut **target="_blank"** akan menampilkan halaman link di dalam jendela baru web browser. Tapi karena saat ini semua web browser menggunakan sistem ‘tab’, maka halaman link umumnya akan tampil di dalam tab baru. Berikut contoh penggunaannya:

```
<a href="en.wikipedia.org/wiki/HTML" target="_blank">HTML di Wikipedia</a>
```



Beberapa sumber menyarankan untuk menggunakan atribut `target="_blank"` untuk semua link yang menuju website luar (external link). Alasannya, agar pengunjung web masih dapat ‘kembali’ ke website kita. Karena jika menggunakan `target="_self"`, halaman web kita akan tertimpa dengan halaman baru tersebut.

Tapi untuk link yang menuju halaman lain dari website yang sama (internal link) sebaiknya menggunakan `target="_self"`, agar web browser pengujung tidak ‘penuh’ dengan tab-tab baru.

Selain `_self` dan `_blank`, atribut target masih memiliki nilai `_parent`, `_top` dan `framename`. Ketiga nilai ini digunakan jika halaman HTML menggunakan frame. Frame adalah fitur HTML untuk ‘memecah’ jendela web browser menjadi beberapa bagian. Saat ini penggunaan frame sudah jarang digunakan dan sudah berstatus *deprecated*, oleh karena itu saya akan melewatkkan pembahasan ini.

Atribut Rel

Atribut `rel` merupakan singkatan dari **relationship**. Atribut ini digunakan untuk menyatakan bagaimana hubungan halaman saat ini dengan halaman yang akan di-link-kan.

Nilai dari atribut ini cukup banyak, beberapa diantaranya: `alternate`, `author`, `bookmark`, `noreferrer`, `noreferrerer`, dan `prefetch`. Atribut `rel` lebih ditujukan kepada ‘robot’ mesin pencari seperti google, bukan untuk pengunjung. Oleh karena itu, kita tidak akan melihat efek langsung dari atribut ini.

Atribut rel relatif jarang digunakan, sehingga saya tidak akan membahas semua nilai dari atribut ini. Akan tetapi terdapat satu nilai atribut rel yang cukup ‘terkenal’, yakni `noreferrer`. Atribut `rel="noreferrer"` digunakan untuk tujuan SEO.

Salah satu point terpenting yang digunakan Google untuk membuat rangking hasil pencarian adalah dengan menilai seberapa banyak sebuah situs dijadikan referensi. Google men-kalkulasinya dengan melihat seberapa banyak **link** yang menuju website tersebut, atau dikenal dengan istilah **backlink**. Apabila sebuah website banyak dijadikan referensi dari situs lain, maka google menganggapnya sebagai website terbaik dan ditampilkan pada posisi teratas.

Situs wikipedia hampir selalu berada pada posisi pertama. Hal ini terjadi karena banyak website lain yang menjadikan wikipedia sebagai sumber dan membuat link ke wikipedia. Konsep ini sangat baik, tetapi ada kalanya kita ingin sebuah link ‘tidak ikut dihitung’ oleh google.

Sebagai contoh, ketika saya membuat artikel di duniaIlkom, kadang saya menggunakan referensi dari situs lain dan membuat link kepada website tersebut. Link ini akan ‘dihitung’ oleh google sebagai *backlink* untuk situs tersebut.

Pada bagian bawah artikel, saya menyediakan kolom komentar. Selain mengisi komentar, pengunjung duniaIlkom juga bisa mengisi biodata lain seperti nama, email dan alamat situs pribadi. Alamat situs ini nantinya akan ditampilkan dan menjadi link kepada situs si pembuat komentar.

Akan tetapi, banyak dari situs-situs ini tidak berkaitan dengan pembahasan (dan beberapa diantaranya menuju ke situs jual beli atau bahkan situs *spam*), sehingga saya tidak ingin google menjadikan link tersebut untuk ikut dihitung sebagai backlink.

Beberapa pakar SEO juga tidak menyarankan membuat link ke situs yang tidak berkaitan dengan materi yang dibahas, bahkan bisa menurunkan ranking situs si pemberi link.

Untuk memberitahu google akan hal ini, kita tinggal menambahkan atribut `rel="nofollow"` pada link tersebut, seperti contoh berikut ini:

```
<a href="http://www.duniaikom.com" rel="nofollow">Situs ane gan</a>
```

Ketika google melihat link dengan atribut `rel="nofollow"`, maka google tidak akan menghitung link tersebut untuk menilai ranking website.



Beberapa sumber juga menyarankan untuk menggunakan atribut `rel="nofollow"` pada seluruh link keluar, agar situs kita tidak ‘membantu’ situs orang lain untuk ‘naik ranking’.

Akan tetapi saya pribadi hanya menggunakan `rel="nofollow"` untuk link ke situs yang tidak berkaitan dengan artikel. Untuk situs sumber yang memang menjadi rujukan artikel, tidak ada salahnya kita ‘ikut membantu’ rangking web tersebut, paling tidak sebagai rasa terimakasih.

6.3 Heading Element (h1 – h6)

Heading element digunakan untuk membuat judul (*heading*) di dalam halaman HTML. Heading element terdiri dari 6 level atau 6 tingkatan, yakni dari `<h1>` sampai dengan `<h6>`.

Berikut adalah contoh tampilan tag `<h1>` sampai dengan tag `<h6>` di dalam web browser:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Link HTML</title>
6 </head>
7 <body>
8   <h1>Judul Artikel dengan h1</h1>
9   <h2>Judul Artikel dengan h2</h2>
10  <h3>Judul Artikel dengan h3</h3>
11  <h4>Judul Artikel dengan h4</h4>
12  <h5>Judul Artikel dengan h5</h5>
13  <h6>Judul Artikel dengan h6</h6>
14 </body>
15 </html>
```



Gambar: Contoh tag <h1> sampai <h6>

Dapat anda lihat, secara default bawaan web browser, seluruh tag heading ditampilkan dalam huruf tebal dan dalam berbagai ukuran font. Tag <h1> tampil dengan ukuran font besar, dan tag <h6> menggunakan ukuran font kecil. Jika ingin mengubah efek ini, bisa dilakukan dengan CSS.

Heading element termasuk ke dalam **block level element** dan akan tampil terpisah di dalam baris baru. Web browser juga akan menambahkan beberapa spasi sebelum dan sesudah heading.

Heading element yang memiliki 6 tingkatan ini mencerminkan kegunaannya untuk membuat struktur judul website. Judul dari sebuah artikel sebaiknya ditulis dengan tag <h1>, sub judul ditulis menggunakan <h2>, sub sub judul dengan <h3>, dst. Struktur yang teratur juga menjadi nilai tambah untuk *search engine*.

Agar struktur menjadi konsisten, usahakan agar tidak ‘melompati’ struktur penulisan. Contoh ‘lompatan’ ini seperti membuat judul dengan <h1>, kemudian langsung menggunakan tag <h4> untuk sub judulnya.

Selain dalam artikel, heading element juga sering dijadikan judul sidebar, judul untuk bagian footer, dan berbagai bagian lain dari website.

Jika anda melihat sebuah judul artikel yang juga berfungsi sebagai link, biasanya hal ini dilakukan dengan menyisipkan tag <a> diantara header element, seperti contoh berikut:

```
<h1>
<a href="http://www.duniaikom.com/tutorial_HTML_dasar.html">
    Belajar Tutorial HTML Dasar
</a>
</h1>
```

6.4 Emphasis, Strong, Bold, dan Italic Element

Keempat element diatas digunakan untuk menghasilkan efek huruf tebal dan huruf miring dari text. Untuk mendapatkan efek huruf miring, kita bisa menggunakan tag `` (*emphasis element*) atau tag `<i>` (*italic element*), sedangkan untuk efek huruf tebal, bisa menggunakan tag `` (*strong importance*) atau tag `` (*bold element*).

Keempat element ini termasuk *inline level element*, sehingga akan mengikuti alur yang ada (tidak tampil pada baris baru). Biasanya tag-tag ini digunakan di dalam paragraf (di dalam tag `<p>`).

Akan tetapi, kenapa masing-masing efek diwakilkan dengan 2 buah element? Ini berawal dari kisah sejarah penggunaan HTML.

Tag `` dan tag `<i>` telah tersedia sejak HTML versi awal. Namun pada era HTML 4.01, keduanya dianggap tidak cocok dan berstatus *deprecated*. Alasannya, kedua tag ini memiliki ‘nama’ dengan efek visual, dan tidak mencerminkan struktur.

Tag `<i>` yang merupakan singkatan dari *italics* dianggap tidak cocok untuk ‘nama’ struktur. W3C berpendapat seharusnya sebuah kata ditulis dengan huruf miring karena kata tersebut perlu penekanan, bukan karena seseorang ingin huruf tersebut tampil miring.

Oleh karena itu, pada HTML 4.01 tag `<i>` dinyatakan *deprecated* dan digantikan dengan tag `` yang merupakan singkatan dari *emphasis* (bahasa Inggris: ‘*penekanan kata*’).



Dalam HTML, tag / element yang mewakili struktur dikenal dengan sebutan **semantic element**. HTML5 membawa banyak tag baru yang memiliki makna semantic. Kita akan mempelajarinya dalam bab tersendiri.

Sama seperti tag `<i>`, tag `` yang merupakan singkatan dari *bold* juga dianggap tidak mencerminkan struktur. Dalam HTML 4.01, tag `` dinyatakan *deprecated* dan diganti dengan tag `` yang memiliki kepanjangan ‘*strong importance*’ (bahasa Inggris: ‘*kata yang penting*’).

Seperti yang anda lihat, alasan dibalik ‘dibuangnya’ tag `<i>` dan tag `` karena ‘salah nama’. Walaupun demikian, web browser tetap mendukung penuh kedua tag ini, dan banyak web developer tetap menggunakan keduanya karena lebih praktis dan singkat.

Peseteruan antara tag `<i>` vs `` dan tag `` vs `` sering menjadi bahan diskusi di setiap kesempatan yang membahas tentang cara memiringkan huruf dengan HTML.

Ketika WHATWG merancang HTML5, alih-alih memilih tag `` dan ``, mereka secara resmi kembali ‘merestui’ penggunaan tag `<i>` dan tag `` (yang dahulu ingin dibuang oleh W3C). Namun kali ini keduanya memiliki fungsi baru.

Dalam standar HTML5, tag `<i>` digunakan untuk menandai kata-kata yang tidak memerlukan penekanan, tetapi ingin ditulis miring, seperti bahasa asing dan istilah teknis. Sedangkan tag `` digunakan untuk menandai kata yang tidak terlalu penting, tapi biasanya ditulis tebal seperti nama produk atau nama merk.

Jadi, tag mana yang digunakan untuk memiringkan atau menebalkan huruf? jawabannya: *terserah* :) Keempat tag ini didukung penuh oleh semua web browser, terlebih lagi HTML5 merestui penggunaan keempatnya.

Tetapi jika anda mengikuti standar HTML5, maka untuk kata-kata yang penting dan ingin ditekankan maknanya, gunakan `` atau ``. Sedangkan untuk kata yang tidak terlalu penting, tapi ingin ditulis miring atau tebal, gunakan `<i>` atau ``.

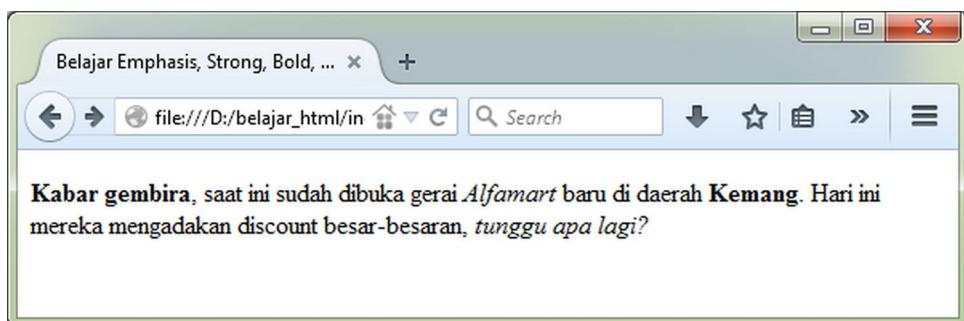
Berikut adalah contoh penggunaan keempat tag ini:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Emphasis, Strong, Bold, dan Italic Element</title>
6 </head>
7 <body>
8   <p>
9     <strong>Kabar gembira</strong>, saat ini sudah dibuka gerai
10    <i>Alfamart</i> baru di daerah <b>Kemang</b>.
11    Hari ini mereka mengadakan discount besar-besaran,
12    <em>tunggu apa lagi?</em>
13  </p>
14 </body>
15 </html>

```



Gambar: Contoh penggunaan tag ``, ``, `<i>` dan ``

6.5 Underline dan Strikethrough Element

Sesuai dengan namanya, **underline element** digunakan untuk membuat efek garis bawah untuk teks, sedangkan **strikethrough element** digunakan untuk membuat efek garis tercoret.

Sama seperti kasus tag `<i>` dengan `` dan tag `` dengan ``, dalam HTML terdapat dua jenis tag untuk masing-masing efek. Untuk membuat garis bawah (*underline*), tersedia tag `<u>` dan tag `<ins>`, sedangkan untuk efek garis tercoret bisa menggunakan tag `<s>` dan tag ``.

Kenapa ada 2 jenis tag? Jawabannya sama seperti pada kasus huruf miring dan huruf tebal.

Pada era HTML 4.01, tag `<u>` dan tag `<s>` dianggap tidak mewakili struktur, sehingga W3C ingin menggantinya dengan tag `<ins>` dan ``. Namun ketika HTML5 datang, keempatnya kembali dianggap valid dan di-sah-kan menjadi standar.

Tag `<ins>` adalah tag yang dahulunya direstui oleh HTML 4.01. Tag ini merupakan singkatan dari **insert**. Sesuai dengan namanya, tag `<ins>` digunakan untuk menandai bagian teks yang ditambahkan kemudian, yakni ketika merevisi sebuah tulisan. Di dalam web browser, teks yang berada di antara tag pembuka `<ins>` dan tag penutup `</ins>` akan ditampilkan dengan efek garis bawah (*underline*).

Tag `<u>` juga akan ditampilkan dengan efek garis bawah. Tag ini telah tersedia sejak awal kehadiran HTML, yang merupakan singkatan dari *underline*. Karena ‘nama’ tag ini yang bermakna tampilan dan bukan struktur, dalam HTML 4.01 penggunaannya tidak disarankan (*deprecated*), tetapi dalam HTML5 tag ini tetap dianggap valid.



Sedapat mungkin anda mempertimbangkan penggunaan efek garis bawah di dalam teks. Teks dengan tanda garis bawah sering dianggap sebagai link, sehingga akan sangat mungkin membuat pengunjung website bingung, karena mereka beranggapan bahwa ini adalah link dan bisa diklik.

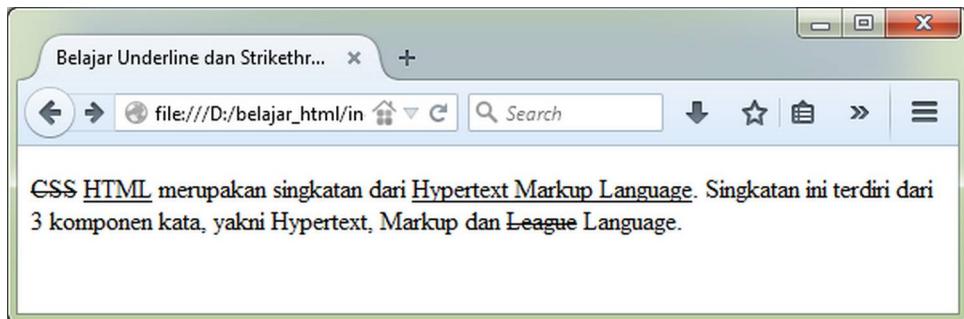
Tag `` merupakan singkatan dari **delete**. Tag ini digunakan untuk menandai teks yang ingin dikoreksi dan tidak lagi relevan. Web browser akan menampilkan teks diantara tag pembuka `` dan tag penutup `` dengan efek garis tercoret. Di dalam HTML 4.01, tag `` digunakan untuk mengganti tag `<s>`.

Tag `<s>` telah tersedia sejak versi awal HTML. Tag `<s>` adalah singkatan dari **strikethrough**. Namun karena ‘nama’-nya juga tidak mencerminkan struktur (lebih ke efek tampilan), penggunaan tag ini tidak direstui oleh HTML 4.01 (*deprecated*), tetapi di dalam HTML5 penggunaannya kembali direstui.

Berikut adalah contoh penggunaan tag `<ins>`, `<u>`, `` dan `<s>`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Underline dan Strikethrough Element</title>
6 </head>
7 <body>
8   <p>
9     <s>CSS</s> <ins>HTML</ins> merupakan singkatan dari
10    <u>Hypertext Markup Language</u>.
11    Singkatan ini terdiri dari 3 komponen kata,
12    yakni Hypertext, Markup dan <del>League</del> Language.
13   </p>
14 </body>
15 </html>
```



Gambar: Contoh penggunaan tag `<ins>`, `<u>`, `` dan `<s>`



Selain menggunakan tag `` dan `<s>`, HTML juga memiliki tag `<strike>` dengan tampilan yang sama, namun HTML5 menyatakan tag `<strike>` deprecated dan sebaiknya tidak digunakan lagi.

6.6 Superscript dan Subscript Element

Superscript adalah sebutan untuk karakter kecil diatas text yang umumnya digunakan di dalam penulisan rumus matematika seperti ‘pangkat’. Dalam angka 3^2 , angka 2 disebut dengan *superscript*. Dalam HTML untuk membuat superscript, digunakan tag `<sup>`.

Subscript adalah penyebutan untuk karakter kecil yang diletakkan sedikit di bawah baris normal. Biasanya subscript digunakan untuk penulisan formula kimia seperti CO_2 . Angka 2 disini dinamakan *subscript*. Untuk membuat subscript, HTML menyediakan tag `<sub>`.

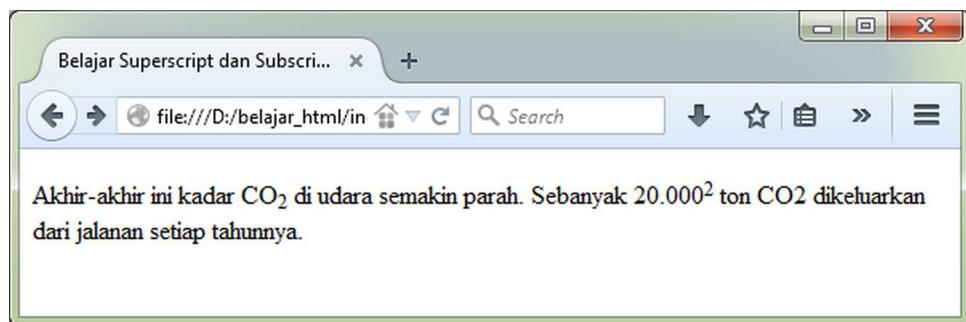
Berikut adalah contoh penggunaan tag `<sup>` dan tag `<sub>`:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Superscript dan Subscript Element</title>
6 </head>
7 <body>
8   <p>
9     Akhir-akhir ini kadar CO<sub>2</sub> di udara semakin parah.
10    Sebanyak 20.000<sup>2</sup> ton CO2 dikeluarkan dari jalanan
11      setiap tahunnya.
12   </p>
13 </body>
14 </html>

```



Gambar: Contoh penggunaan tag <sup> dan tag <sub>

6.7 Preformatted Element

Tag <pre> merupakan singkatan dari **preformatted**, yang berarti *tidak diformat*. Sesuai dengan namanya, teks yang berada diantara tag <pre> dan tag </pre> akan ditampilkan ‘apa adanya’, termasuk karakter *whitespace* seperti spasi, tab, dan enter (*carriage returns*).

Preformatted element akan ditampilkan oleh web browser dalam font khusus dimana panjang seluruh huruf sama (*fixed-width font*) biasanya font yang digunakan adalah **Courier New**, sehingga cocok digunakan untuk menampilkan kode program atau teks yang bersifat teknis lainnya.

Tag <pre> termasuk ke dalam jenis *block level element*.

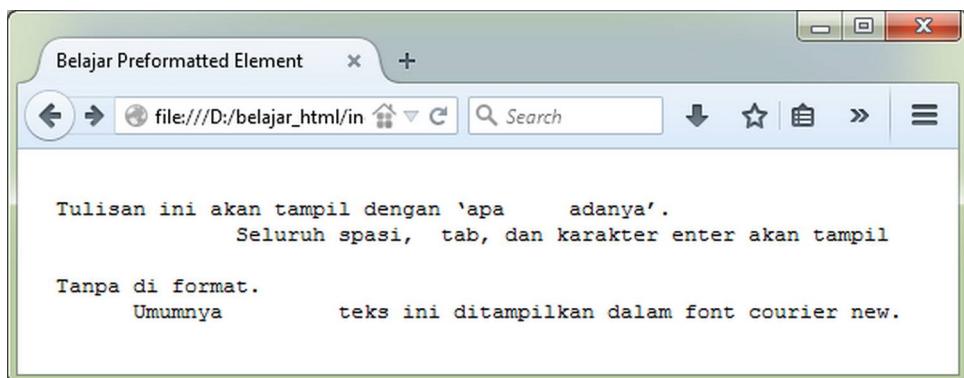


Di dalam HTML, font seperti Courier New ini termasuk ke dalam kelompok font yang disebut dengan **monospace**.

Berikut adalah contoh penggunaan tag <pre>:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Preformatted Element</title>
6 </head>
7 <body>
8   <pre>
9     Tulisan ini akan tampil dengan ‘apa      adanya’ .
10    Seluruh spasi,          tab, dan karakter enter akan tampil
11
12   Tanpa di format.
13     Umumnya          teks ini ditampilkan dalam font courier new.
14   </pre>
15 </body>
16 </html>
```



Gambar: Contoh penggunaan tag <pre>

Pada penulisan teks dalam HTML, biasanya web browser akan mengabaikan spasi kedua dan seterusnya. Misalkan anda membuat <p>kata dengan beberapa spasi</p>, maka yang tampil adalah “kata dengan beberapa spasi”. Kecuali teks tersebut berada di dalam tag khusus seperti <pre>, web browser hanya akan menggunakan 1 karakter spasi saja,

6.8 Code, Samp, KBD, dan Var Element

Keempat element ini mungkin akan jarang digunakan. Saya membahas tag-tag ini karena sifatnya mirip dengan tag <pre>. Keempatnya digunakan untuk membuat struktur kode-kode teknis.

Tag <code> merupakan singkatan dari *code program*. Tag <samp> singkatan dari *sample program*, tag <kbd> singkatan dari *keyboard input*, dan tag <var> singkatan dari *variable*.

Seperti yang anda lihat, keempat tag ini digunakan dalam struktur teks teknis, khususnya kode-kode program komputer. Keempatnya juga termasuk *inline level element*, sehingga akan mengikuti baris dimana tag ini dituliskan.

Tag <code>, <samp> dan <kbd> akan tampil dengan font *monospace* seperti tag <pre>, sedangkan tag <var> ditampilkan dengan huruf miring.

Berbeda dengan tag <pre>, keempat tag ini akan memformat *whitespace*, yang berarti jika anda membuat 2 spasi atau lebih, web browser hanya menggunakan 1 spasi saja.

Berikut adalah contoh penggunaan keempat element ini:

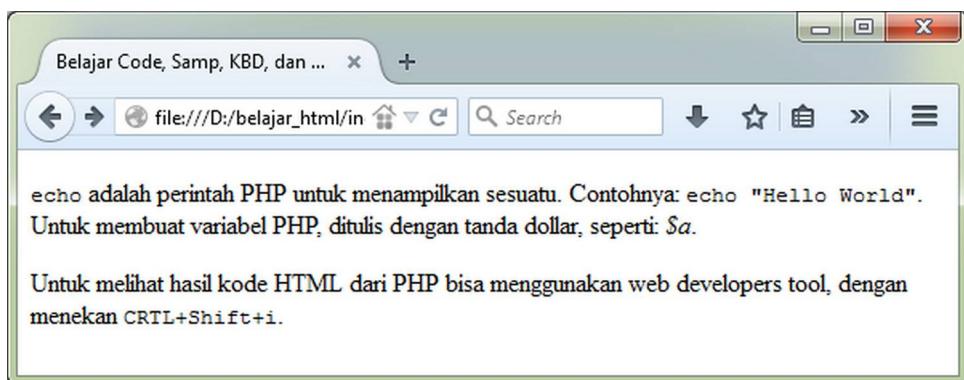
index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Code, Samp, KBD, dan Var Element</title>
6 </head>
7 <body>
8   <p>
9     <code>echo</code> adalah perintah PHP untuk menampilkan sesuatu.

```

```
10 Contohnya: <samp>echo "Hello World"</samp>.  
11 Untuk membuat variabel PHP, ditulis dengan tanda dollar,  
12 seperti: <var>$a</var>.  
13 </p>  
14 <p>  
15 Untuk melihat hasil kode HTML dari PHP bisa menggunakan  
16 web developers tool, dengan menekan <kbd>CTRL+Shift+i</kbd>.  
17 </p>  
18 </body>  
19 </html>
```



Gambar: Contoh penggunaan tag `<code>`, `<samp>`, `<kbd>` dan `<var>`

6.9 Cite, Quote dan Blockquote Element

Ketiga element yang akan kita bahas ini digunakan untuk membuat kutipan.

Tag `<cite>` adalah singkatan dari *citation* yang digunakan untuk menandai hasil karya seseorang seperti judul buku, judul lagu, judul film, musisi, dll. Tag `<cite>` akan ditampilkan dengan huruf miring dalam web browser.

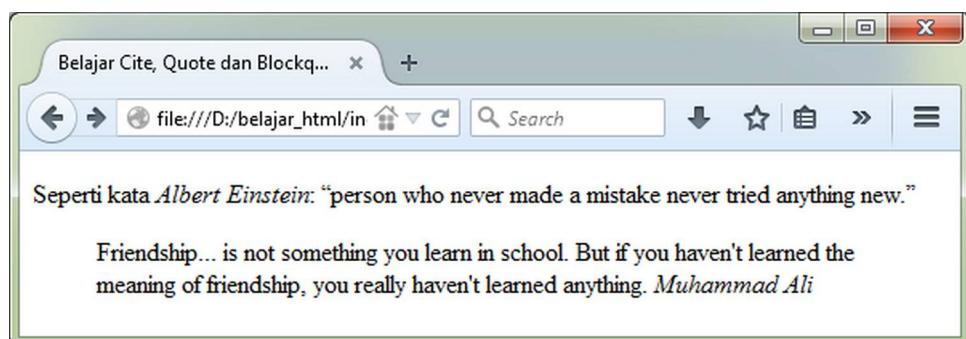
Tag `<q>` merupakan singkatan dari *quote* yang berarti *kutipan*. Tag ini digunakan untuk membuat kutipan singkat. Di dalam web browser, teks diantara tag pembuka `<q>` dan tag penutup `</q>` akan tampil seperti teks biasa dengan penambahan tanda kutip pada awal dan akhir teks.

Tag `<blockquote>` juga digunakan untuk membuat kutipan, tetapi berbeda dengan tag `<q>` yang merupakan *inline level element*, tag `<blockquote>` adalah *block level element*, sehingga akan tampil dalam baris baru, terpisah dengan baris sebelumnya. Web browser akan menambahkan beberapa spasi (*margin*) di sisi kiri teks untuk tag `<blockquote>`. Jika tag `<q>` digunakan untuk membuat kutipan singkat, tag `<blockquote>` cocok untuk membuat kutipan panjang.

Berikut adalah contoh penggunaan tag `<cite>`, `<q>`, dan `<blockquote>`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Cite, Quote dan Blockquote Element</title>
6 </head>
7 <body>
8   <p>
9     Seperti kata <cite>Albert Einstein</cite>:
10    <q>person who never made a mistake never tried anything new.</q>
11  </p>
12  <blockquote>
13    Friendship... is not something you learn in school.
14    But if you haven't learned the meaning of friendship,
15    you really haven't learned anything.
16    <cite>Muhammad Ali</cite>
17  </blockquote>
18 </body>
19 </html>
```



Gambar: Contoh penggunaan tag <cite>, <q>, dan <blockquote>

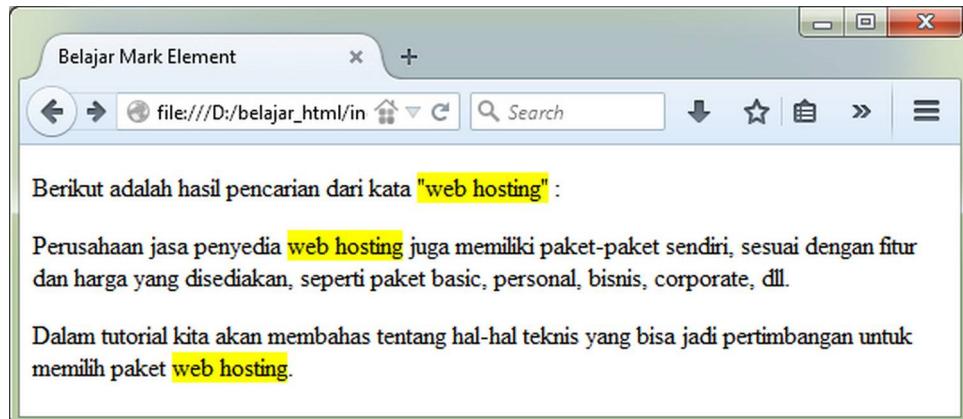
6.10 Mark Element

Tag <mark> adalah salah satu tag baru yang diperkenalkan pada HTML5. Tag ini berfungsi untuk menandai bagian teks yang dianggap penting, relatif kepada teks di sekelilingnya. Tag ini juga bisa digunakan untuk menandai hasil pencarian.

Berikut contoh penggunaan tag <mark>:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Mark Element</title>
6 </head>
7 <body>
8   <p>
9     Berikut adalah hasil pencarian dari kata <mark>"web hosting"</mark> :
10  </p>
11  <p>
12    Perusahaan jasa penyedia <mark>web hosting</mark>
13    juga memiliki paket-paket sendiri, sesuai dengan fitur
14    dan harga yang disediakan, seperti paket basic, personal,
15    bisnis, corporate, dll.
16  </p>
17  <p>
18    Dalam tutorial kita akan membahas
19    tentang hal-hal teknis yang bisa jadi pertimbangan
20    untuk memilih paket <mark>web hosting</mark>.
21  </p>
22 </body>
23 </html>
```



Gambar: Contoh penggunaan tag <mark>

Seperti yang terlihat, web browser akan menampilkan teks yang di-*mark* dengan warna background kuning.

6.11 Ruby Element

Tag <ruby> digunakan untuk membuat karakter khusus yang digunakan pada bahasa asia timur seperti Jepang, China, atau Korea. Kata ‘ruby’ sendiri tidak ada hubungannya dengan batu mulia

‘ruby’, tapi berasal dari istilah bahasa inggris yang merujuk kepada karakter ‘furigana’ dalam bahasa Jepang.

Tag `<ruby>` adalah tag baru dari HTML5, dengan ‘tag pembantu’ `<rt>` (ruby text) dan `<rp>` (ruby parentheses). Jika anda tidak menulis karakter jepang, china, atau korea, tag ini tidak akan diperlukan.

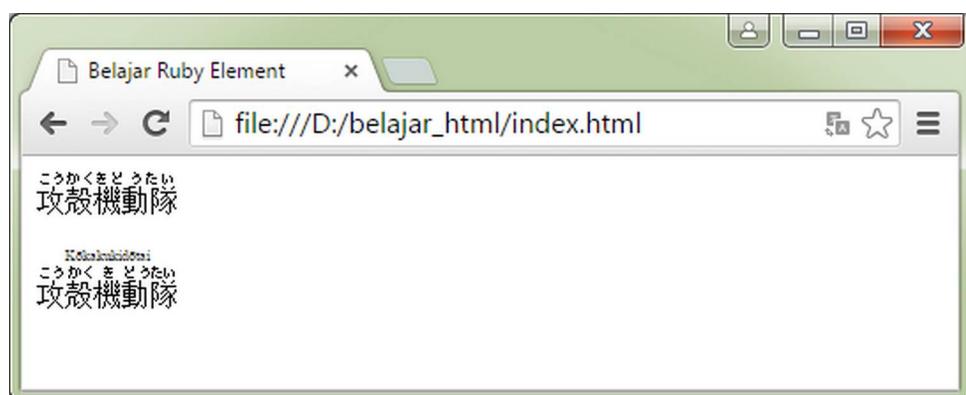
Karena karakter jepang tidak bisa tampil di versi pdf/epub ini, saya akan tampilkan dalam bentuk gambar (kode programnya bisa anda lihat di file `belajar_html.zip`):

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Belajar Ruby Element</title>
6  </head>
7  <body>
8      <ruby>
9          攻殲<rt>こうかく</rt>機動隊<rt>きどうたい</rt>
10     </ruby>
11     <br><br>
12     <ruby>
13         攻<rp>(</rp><rt>こう</rt><rp>)</rp>
14         殲<rp>(</rp><rt>かく</rt><rp>)</rp>
15         機<rp>(</rp><rt>き</rt><rp>)</rp>
16         動<rp>(</rp><rt>どう</rt><rp>)</rp>
17         隊<rp>(</rp><rt>たい</rt><rp>)</rp>
18         <rp>(</rp><rt>Kōkakukidōtai</rt><rp>)</rp>
19     </ruby>
20 </body>
21 </html>

```

Gambar: Kode HTML untuk contoh tag `ruby`



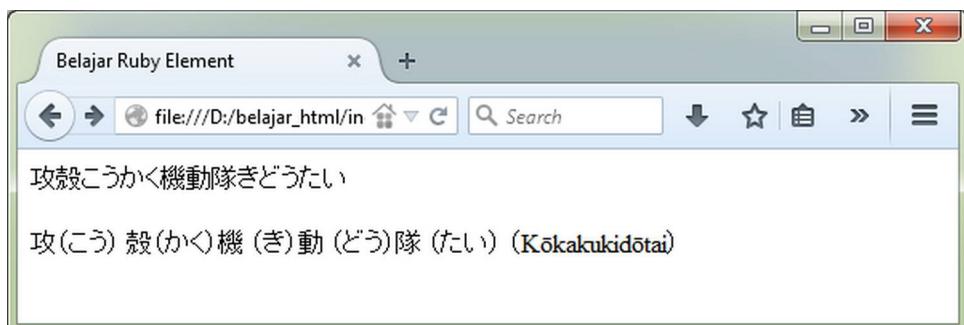
Gambar: Contoh penggunaan tag `<ruby>`, `<rt>` dan `<rp>`



Agar kode diatas bisa berjalan, kita harus mengatur format text dengan UTF-8. Jika menggunakan Notepad++ bisa ke menu **Setting->Preferences->New Document**, dari jendela yang tampil pilih **UTF-8 without BOM** pada bagian encoding (cek juga *apply to opened ANSI files*), kemudian klik **Close**.

Tag `<rt>` berfungsi ‘menaikkan’ karakter. Dalam aksara jepang, huruf yang berada di atas seperti ini digunakan untuk pengucapan dari karakter kanji yang ada dibawahnya.

Tag `<rp>` berfungsi menambahkan tanda kurung. Tanda kurung ini akan tampil untuk web browser yang tidak mendukung tag `<ruby>`. Tag `<rp>` akan memisahkan bagian ‘kanji’ dan ‘pengucapan’.



Gambar: Tampilan tag `<ruby>` di Mozilla Firefox

Tampilan diatas adalah hasil kode program sebelumnya dari web browser yang belum mendukung tag `<ruby>`, perhatikan penambahan tanda kurung.

6.12 Abbr Element

Tag `<abbr>` merupakan singkatan dari **Abbreviation**, yang berarti singkatan. Tag ini digunakan untuk menandai sebuah singkatan di dalam artikel.

Atribut yang sering digunakan untuk tag `<abbr>` adalah **title**. Atribut **title** berfungsi untuk menampilkan kepanjangan dari singkatan yang ada di dalam tag `<abbr>`. Agar kepanjangan yang ditulis dalam atribut ini dapat dilihat, kita harus menempatkan mouse diatas teks tersebut (*mouse over*).

Berikut adalah contoh penggunaan tag `<abbr>`:

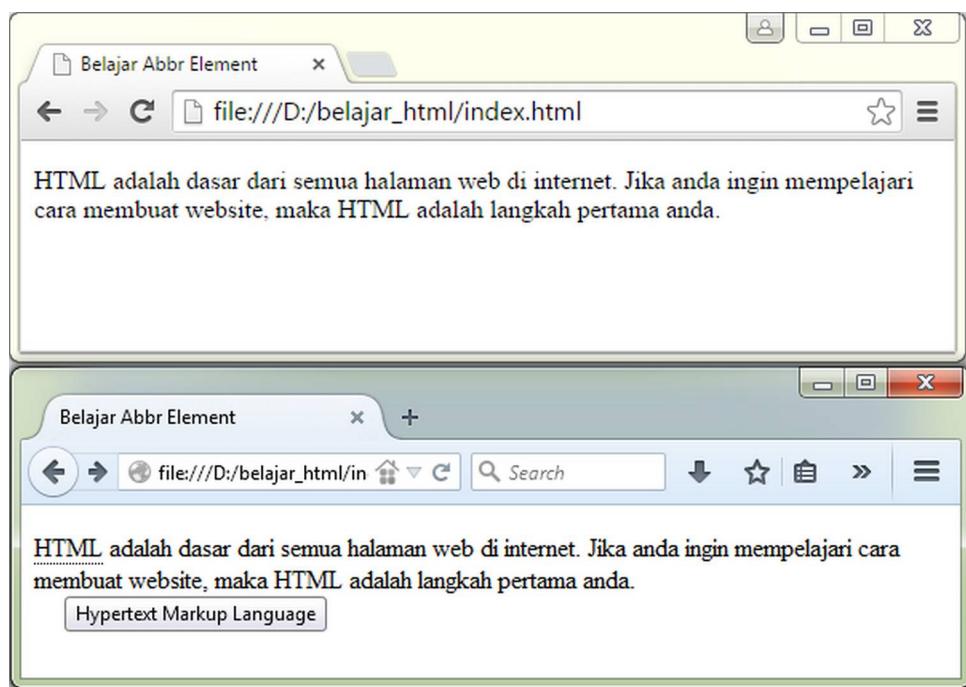
index.html

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Belajar Abbr Element</title>
6  </head>
7  <body>
8      <p>
```

```

9   <abbr title="Hypertext Markup Language">HTML</abbr>
10  adalah dasar dari semua halaman web di internet.
11  Jika anda ingin mempelajari cara membuat website,
12  maka <abbr>HTML</abbr> adalah langkah pertama anda.
13  </p>
14 </body>
15 </html>
```



Gambar: Tampilan tag `<abbr>` di Chrome (atas) dan Firefox (bawah)

Tag `<abbr>` ditampilkan beragam oleh web browser. Web browser *Mozilla Firefox* dan *Opera* akan menampilkan teks dengan garis putus-putus dibawah teks. Tampilan ini baru terlihat jika tag `<abbr>` menggunakan atribut `title`. Di dalam *Google Chrome*, tag `<abbr>` terlihat seperti teks biasa (tanpa garis bawah) dengan atau tanpa atribut `title`.



Title merupakan *global attribut* yang bisa digunakan untuk hampir semua tag lain (tidak hanya untuk tag `<abbr>` saja).

6.13 Address Element

Tag `<address>` digunakan untuk konten yang berisi alamat dan merupakan *block level element*. Umumnya web browser menampilkan tag ini dengan huruf miring (*italic*).

Tag `<address>` bisa anda tempatkan di bagian bawah sebuah artikel (alamat untuk menghubungi penulis artikel), di bagian *footer*, atau halaman tersendiri, seperti halaman *contact us*.

Berikut adalah contoh penggunaan tag `<address>`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Address Element</title>
6 </head>
7 <body>
8   <address>
9     Anda bisa menghubungi kami di
10    <a href="http://www.duniailkom.com/contact_us">
11      www.duniailkom.com</a> .
12    <br>
13    Jika ada kritik/saran/pertanyaan, bisa email ke
14    <a href="mailto:admin@duniailkom.com">admin duniailkom</a> .
15    <br>
16    Alamat kantor kami berada di: Jl. Sudirman Blok 37,
17    No 42A, Pegangsaan Dua, Jakarta Utara. 14250.
18  </address>
19 </body>
20 </html>
```



Gambar: Contoh penggunaan tag `<address>`

6.14 Bdo Element

Tag `<bdo>` merupakan singkatan dari **bidirectional override**. Tag ini ditujukan untuk merubah arah text yang normalnya ditulis dari kiri ke kanan seperti bahasa inggris atau bahasa indonesia, menjadi dari kanan ke kiri seperti penulisan dalam bahasa arab.

Untuk menjalankan fungsinya, tag `<bdo>` memerlukan sebuah atribut **dir** yang bisa berisi salah satu dari 2 nilai, yakni **ltr** dan **rtl**.

Nilai Atribut **Ltr** adalah singkatan dari **left to right** yang akan membuat arah text mulai dari arah kiri lalu ke arah kanan web browser. Sedangkan **rtl** adalah singkatan dari **right to left** yang akan menampilkan text dari kanan ke kiri.

Tag `<bdo>` termasuk ke dalam tipe tag inline, dan akan ditampilkan mengikuti alur text yang ada.

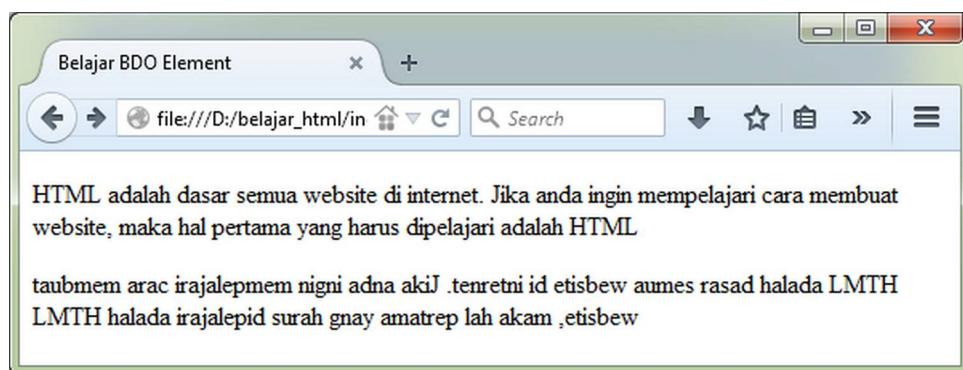
Berikut contoh cara penulisan dan penggunaan tag `<bdo>` di dalam HTML:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar BDO Element</title>
6 </head>
7 <body>
8   <p>
9     <bdo dir="ltr">HTML adalah dasar semua website di internet.
10    Jika anda ingin mempelajari cara membuat website,
11      maka hal pertama yang harus dipelajari adalah HTML</bdo>
12  </p>
13  <p>
14    <bdo dir="rtl">HTML adalah dasar semua website di internet.
15    Jika anda ingin mempelajari cara membuat website,
16      maka hal pertama yang harus dipelajari adalah HTML</bdo>
17  </p>
18 </body>
19 </html>

```



Gambar: Contoh penggunaan tag `<bdo>`

6.15 Dfn Element

Tag `<dfn>` merupakan singkatan dari **definition**. Tag ini digunakan untuk menandai definisi dari sesuatu. Sama seperti tag `<abbr>`, tag `<dfn>` juga sering ditambahkan atribut `title` untuk menerangkan definisi tersebut, atau digunakan bersama-sama dengan tag `<abbr>` (jika definisi tersebut juga merupakan singkatan).

Dalam web browser, tag `<dfn>` ditampilkan dengan huruf miring (*italic*). Berikut adalah contoh penggunaan tag `<dfn>`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Dfn Element</title>
6 </head>
7 <body>
8   <p>
9     <dfn>HTML</dfn> adalah bahasa markup standar
10    yang digunakan untuk membuat halaman web.
11  </p>
12  <p>
13    <dfn title="Hypertext Markup Language">HTML</dfn>
14    adalah bahasa markup standar yang digunakan
15    untuk membuat halaman web.
16  </p>
17  <p>
18    <dfn>
19      <abbr title="Hypertext Markup Language">HTML</abbr>
20    </dfn>
21    adalah bahasa markup standar yang digunakan
22    untuk membuat halaman web.
23  </p>
24 </body>
25 </html>
```



Gambar: Contoh penggunaan tag <dfn>

Dalam contoh diatas, anda dapat melihat 3 alternatif cara penggunaan tag <dfn>.

6.16 Small Element

Tag <small> termasuk kepada salah satu tag *deprecated* di dalam HTML 4.01, kemudian ‘dibangkitkan’ kembali oleh HTML5.

Teks yang berada di dalam tag `<small>` ditampilkan lebih kecil dibandingkan ukuran teks normal. Karena sifatnya yang lebih kepada tampilan inilah, tag `<small>` ingin dihapus oleh W3C pada HTML 4.01.

Dalam HTML5, tag `<small>` di definisikan ulang untuk menandai bagian teks yang bukan bagian dari konten, seperti tulisan *copyright*, *disclaimer* atau *legal notice*.

Berikut adalah contoh penggunaan tag `<small>`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Small Element</title>
6 </head>
7 <body>
8   <p>
9     DuniaIlkom adalah situs belajar ilmu komputer,
10    diharapkan duniaIlkom.com akan dapat menjadi
11    media belajar dan saling berbagi tentang programming,
12    hardware, maupun toeri seputar ilmu komputer.
13    <small>2015 Dunia Ilkom. All Rights Reserved.</small>
14  </p>
15 </body>
16 </html>
```



Gambar: Contoh penggunaan tag `<small>`

6.17 Wbr Element

Tag `<wbr>` relatif baru dan diperkenalkan oleh HTML5. Tag `<wbr>` merupakan singkatan dari **Word Break Opportunity**, yang berarti: ‘*penanda pemisah kata*’. Tag ini digunakan untuk memberitahu web browser dimana boleh dilakukan pemotongan kata.

Efek ini baru akan terlihat jika sebuah kata panjang ditempatkan dalam ‘container’ yang kecil, atau ketika jendela web browser di perkecil. Agar lebih mudah dipahami, perhatikan tampilan berikut ini:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Wbr Element</title>
6 </head>
7 <body>
8   <h2>Belajar HTML</h2>
9   <p>
10    Baiklah, saya bersedia untuk mempertanggungjawabkan
11    perbuatan yang saya lakukan di hari itu.
12    Kejadian itu berawal dari hal-hal sepele.
13  </p>
14 </body>
15 </html>
```



Gambar: Teks ‘mempertanggungjawabkan’ pindah ke baris baru

Seperti yang anda lihat dari tampilan diatas, kata “*mempertanggungjawabkan*” terlalu panjang, sehingga web browser menampilkannya di baris kedua, padahal baris pertama masih bersisa namun tidak cukup ruang bagi kata tersebut untuk bisa naik ke baris pertama.

Salah satu solusi untuk hal ini adalah dengan memberi tanda hubung. Kata “mempertanggungjawabkan” bisa kita potong menjadi “memper-tanggung-jawabkan”. Dengan demikian web browser akan memisahkan kata tersebut pada karakter “-“, seperti tampilan dibawah ini:



Gambar: Solusi 1: ‘memper-tanggung-jawabkan’

Namun, karakter “-” akan tetap ditampilkan apabila terdapat ruang yang cukup untuk menampilkan seluruh kata.

Cara paling baik adalah dengan menyiapkan tag `<wbr>` untuk menggantikan karakter “-”, seperti berikut ini:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Wbr Element</title>
6 </head>
7 <body>
8   <h2>Belajar HTML</h2>
9   <p>
10    Baiklah, saya bersedia untuk memper<wbr>tanggung<wbr>jawabkan
11    perbuatan yang saya lakukan di hari itu.
12    Kejadian itu berawal dari hal-hal sepele.
13  </p>
14 </body>
15 </html>

```



Gambar: Solusi 2: ‘mempertanggungjawabkan’

Seperti yang terlihat, Web browser akan menampilkannya dalam 1 kata jika jendela web browser cukup lebar, dan memecah kata tersebut jika jendela web browser tidak cukup lebar.

Web browser akan memecah kata pada bagian yang ditandai tag `<wbr>` sehingga tampilan teks tetap rapi. Penggunaan tag `<wbr>` cocok jika web anda menggunakan desain responsive dan memiliki kata-kata yang panjang.

6.18 Deprecated Element dan Atribut

Selama perjalannya dari HTML versi awal hingga HTML5, terdapat beberapa tag yang sudah usang atau dikenal dengan istilah **deprecated**. Tag-tag ini mungkin masih akan anda jumpai, karena beberapa web browser masih mendukungnya (dengan alasan kompatibilitas untuk website ‘jadul’).

Berikut adalah beberapa tag-tag deprecated di dalam HTML, yang sebaiknya tidak digunakan lagi:

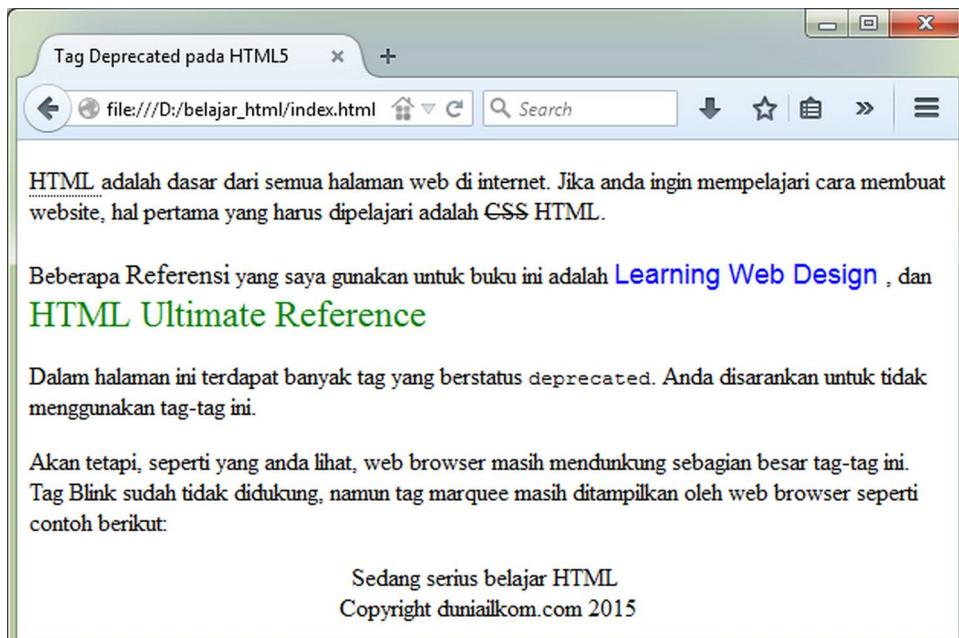
- Tag `<acronym>`: untuk membuat singkatan. Tag ini digantikan dengan tag `<abbr>`.
- Tag `<applet>`: untuk memasukkan Java Applet.
- Tag `<basefont>`: untuk membuat settingan font default halaman web. Untuk merubah tampilan font, sebaiknya menggunakan CSS.
- Tag `<big>`: untuk membuat ukuran text menjadi besar. Tag ini murni untuk tampilan dan disarankan menggunakan CSS.
- Tag `<center>`: untuk membuat rata tengah text. Disarankan untuk menggunakan CSS, namun tag ini masih banyak digunakan karena lebih praktis.
- Tag `<dir>`: untuk membuat directory list. Tag ini digantikan dengan tag `` (unordered list).
- Tag ``: Untuk mengatur font text. Tag ini masih sering digunakan untuk mengatur tampilan text, seperti warna, ukuran font, jenis font, dll. Tag ini sepenuhnya untuk mengatur tampilan dan disarankan menggunakan CSS.
- Tag `<isindex>`: untuk memasukkan search box.
- Tag `<strike>`: untuk membuat text dengan garis salah atau strike-through. Tag ini digantikan dengan tag `<s>` atau ``.
- Tag `<tt>`: untuk membuat text dengan tampilan *teletype* dan ditampilkan dengan font *monospace*. Tag ini bisa digantikan dengan tag `<code>`.
- Tag `<blink>`: Tag ini berasal dari era *browser war* antara *Internet Explorer* dan *Netscape*. Tag `<blink>` digunakan untuk membuat text berkedip-kedip yang cukup fenomenal pada zamannya. Saat ini penggunaan tag `<blink>` dianggap mengganggu pengunjung dan web browser sudah tidak mendukungnya.
- Tag `<marquee>`: Tag ini juga berasal dari era *browser war* antara IE dan Netscape. Tag `<marquee>` digunakan untuk membuat efek text berjalan pada web browser. Tag ini adalah sedikit tag HTML yang menghasilkan efek ‘bergerak’. Beberapa web browser masih mendukung tag `<marquee>`.
- Tag `<frame>` dan `<frameset>`: Kedua tag ini digunakan untuk menampilkan beberapa halaman html di sebuah layar tampilan yang sama. Saat ini penggunaan frame sudah tidak disarankan lagi, walaupun web browser modern masih mendukungnya.

Jika anda ingin melihat apakah web browser masih mendukung tag-tag ini, silahkan jalankan contoh kode HTML berikut ini:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Tag Deprecated pada HTML5</title>
6 </head>
7 <body>
8   <p>
9     <acronym title="HyperText Markup Language">
10       HTML
11     </acronym>
12     adalah dasar dari semua halaman web di internet.
13     Jika anda ingin mempelajari cara membuat website,
14     hal pertama yang harus dipelajari adalah
15     <strike>CSS</strike> HTML.
16   </p>
17
18   <p>
19     Beberapa <big>Referensi</big>
20     yang saya gunakan untuk buku ini adalah
21     <font color="blue" face="arial" size="4px">
22       Learning Web Design
23     </font>,
24     dan <font color="green" size="5px">
25       HTML Ultimate Reference
26     </font>
27   </p>
28
29   <p>
30     Dalam halaman ini terdapat banyak tag yang berstatus
31     <tt>deprecated</tt>.
32     Anda disarankan untuk tidak menggunakan tag-tag ini.
33   </p>
34
35   <p>
36     Akan tetapi, seperti yang anda lihat,
37     web browser masih mendukung
38     sebagian besar tag-tag ini.
39     <blink>Tag Blink </blink>sudah tidak didukung,
40     namun tag marquee masih ditampilkan oleh web browser
41     seperti contoh berikut:
42   </p>
43
44   <marquee>Sedang serius belajar HTML</marquee>
```

```
45 <center>Copyright duniaikom.com 2015</center>
46 </body>
47 </html>
```



Gambar: Contoh hasil tag-tag deprecated di Mozilla Firefox

Seperti yang anda lihat, hampir semua tag deprecated ini masih di dukung oleh web browser. Karena alasan ini pula spesifikasi HTML5 tidak menggunakan istilah **deprecated**, tetapi memilih istilah **obsolete**. Menurut saya, makna kata *obsolete* terdengar lebih sopan dari *deprecated*. Namun keduanya tetap merujuk kepada tag dan atribut yang sebaiknya tidak anda gunakan lagi.

Selain deprecated element, terdapat juga deprecated atribut yang sebaiknya tidak digunakan lagi. Beberapa diantaranya adalah: `bgcolor`, `color`, `marginbottom`, `align`, `border`, `cellpadding`, `cellspacing`, dan `valign`.

Untuk list lengkap mengenai deprecated/obsolete tag dan atribut ini, silahkan mengunjungi situs [WHATWG](#)¹.

6.19 HTML Entity

Dalam HTML, terdapat beberapa karakter khusus yang tidak bisa ditulis untuk ditampilkan ke dalam web browser. Contohnya, kita tidak bisa menulis teks berikut di dalam HTML:

Di dalam HTML, tag `<p>` digunakan untuk membuat paragraf.

¹<https://developers.whatwg.org/obsolete.html>

Web browser akan memproses “<p>” sebagai tag HTML, bukan bagian dari teks. Ini terjadi karena karakter “<” dan “>” memiliki arti khusus di dalam HTML. Untuk mengatasi hal ini, kita harus menggunakan **HTML Entity**.

HTML entity adalah cara penulisan karakter dengan menggunakan simbol khusus agar tidak diterjemahkan oleh web browser sebagai bagian dari HTML. Kita juga bisa menggunakan HTML entity untuk menulis karakter-karakter yang tidak ada di keyboard seperti lambang copyright(©), lambang trademark (™), atau karakter mata uang seperti euro (€).

Terdapat 2 cara Penulisan HTML entity, yakni dengan menggunakan **nama** (disebut juga dengan **named entity**) dan menggunakan **angka** (**numeric entity**). Numeric entity bisa ditulis menggunakan angka desimal (basis 10) atau angka hexadesimal (basis 16).

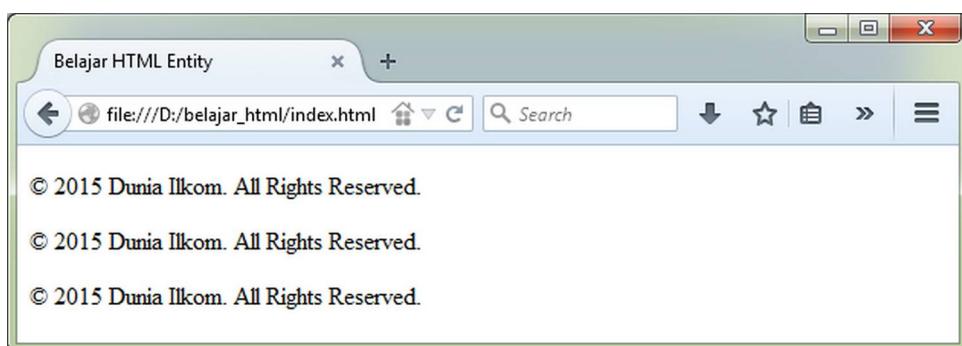
Untuk penulisan HTML entity, diawali dengan karakter “&” dan diakhiri dengan karakter “;”. Sebagai contoh, untuk penulisan karakter copyright, berikut adalah cara penulisannya:

- Named entity: ©
- Numeric entity (decimal): ©
- Numeric entity (hexadecimal): &x000A9;

Langsung saja kita coba menggunakanya di dalam HTML:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar HTML Entity</title>
6 </head>
7 <body>
8   <p>&copy; 2015 Dunia Ilkom. All Rights Reserved.</p>
9   <p>&#169; 2015 Dunia Ilkom. All Rights Reserved.</p>
10  <p>&x000A9; 2015 Dunia Ilkom. All Rights Reserved.</p>
11 </body>
12 </html>
```



Gambar: Contoh penulisan HTML entity

Dapat dilihat bahwa ketiganya ditampilkan dengan karakter copyright. Untuk daftar lengkap mengenai karakter-karakter entity, anda bisa mengunjungi dev.w3.org².

Non-Breaking Space

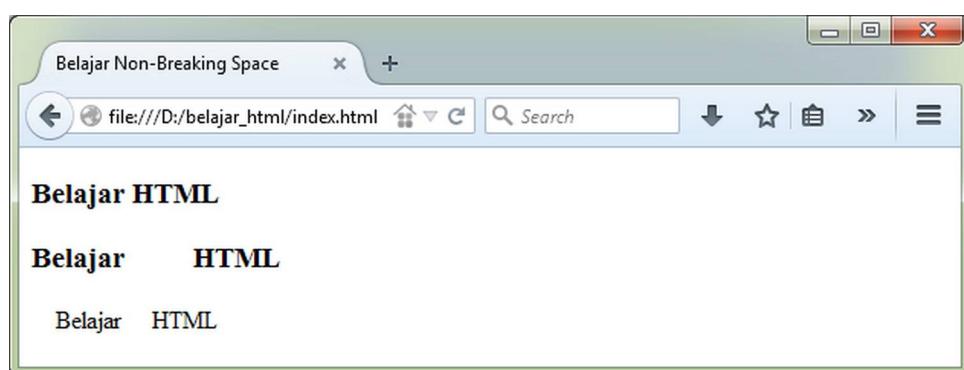
Salah satu karakter HTML Entity yang cukup sering digunakan adalah karakter ‘spasi’, atau dikenal dengan ‘non-breaking space’. Karakter ini memiliki nilai entity: . Non-breaking space sering digunakan jika anda ingin membuat beberapa spasi dalam teks HTML.

Seperti yang telah kita ketahui, jika kita memisahkan kata dengan beberapa spasi, web browser hanya akan menggunakan 1 karakter spasi saja. Jika anda menginginkan kata tersebut dipisah dengan 2 atau lebih spasi, maka harus menggunakan .

Berikut contohnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Non-Breaking Space</title>
6 </head>
7 <body>
8   <h3>Belajar HTML</h3>
9   <h3>Belajar &nbsp; &nbsp; &nbsp; HTML</h3>
10  <p>&nbsp; &nbsp; Belajar &nbsp; &nbsp; HTML</p>
11 </body>
12 </html>
```



Gambar: Contoh penggunaan

Sesuai dengan namanya, kata yang dipisah dengan karakter akan dianggap oleh web browser sebagai satu bagian, sehingga tidak akan terpisah jika ukuran jendela web browser diperkecil (kebalikan dari tag <wbr>).

²<http://dev.w3.org/html5/html-author/charref>

Dalam bab ini kita telah membahas berbagai tag dan atribut yang digunakan untuk memformat teks di dalam HTML. Akan tetapi masih terdapat 1 kelompok tag lagi yang juga digunakan untuk memproses teks, yakni daftar/list element.

Dalam bab berikutnya, kita akan membahas secara detail tentang List Element di dalam HTML.

7. List Element

Pada bab kali ini kita akan membahas cara membuat daftar, atau lebih dikenal dengan istilah **list**. Dalam HTML, terdapat 3 jenis list, yakni **ordered list**, **unordered list**, dan **description list**.

7.1 Ordered List

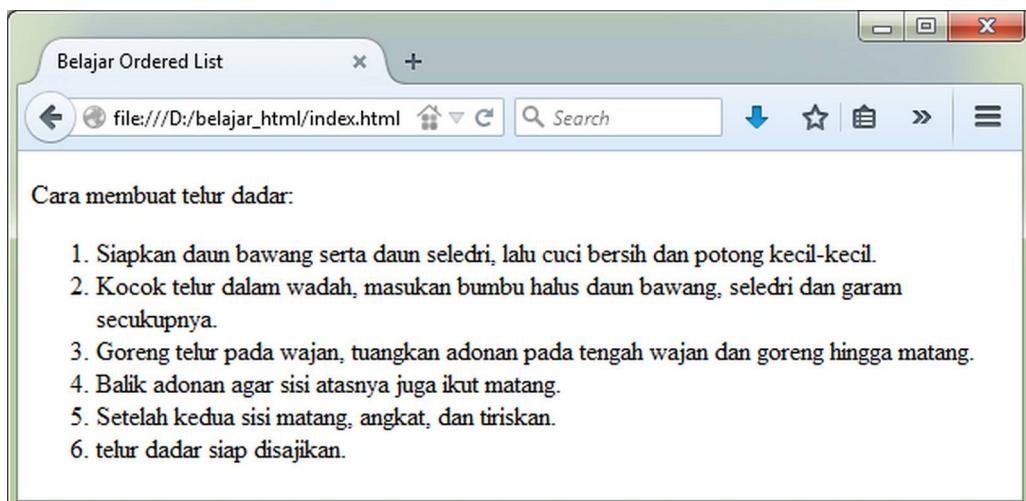
Ordered list adalah list yang berurutan. List ini menggunakan angka atau huruf sebagai penentu urutan. Ordered list cocok digunakan untuk daftar yang harus tersusun berdasarkan kronologis seperti instruksi atau daftar kegiatan.

Untuk membuat ordered list, kita menggunakan tag ``, sedangkan untuk membuat isi dari list, menggunakan tag ``.

Berikut contoh penggunaan ordered list:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Ordered List</title>
6 </head>
7 <body>
8   <p>Cara membuat telur dadar: </p>
9   <ol>
10    <li>Siapkan daun bawang serta daun seledri, lalu cuci bersih
11      dan potong kecil-kecil.</li>
12    <li>Kocok telur dalam wadah, masukan bumbu halus daun bawang,
13      seledri dan garam secukupnya.</li>
14    <li>Goreng telur pada wajan, tuangkan adonan pada tengah wajan
15      dan goreng hingga matang.</li>
16    <li>Balik adonan agar sisi atasnya juga ikut matang.</li>
17    <li>Setelah kedua sisi matang, angkat, dan tiriskan.</li>
18    <li>telur dadar siap disajikan.</li>
19 </ol>
20 </body>
21 </html>
```



Gambar: Contoh penggunaan ordered list

Secara default, *ordered list* ditampilkan dengan urutan angka 1, 2, 3, dst. Namun kita bisa mengubahnya menjadi huruf atau angka romawi menggunakan atribut **type**.

Atribut Type

Atribut **type** pada tag berfungsi untuk menentukan jenis karakter yang digunakan sebagai penanda list. Terdapat 5 jenis type yang tersedia di dalam HTML:

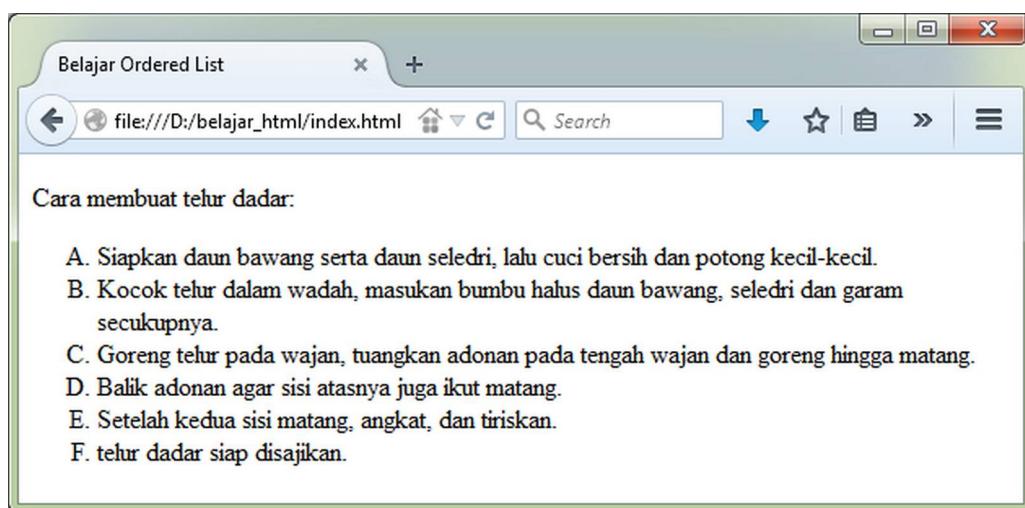
- type="a", menggunakan huruf kecil sebagai penanda list.
- type="A", menggunakan huruf besar sebagai penanda list.
- type="i", menggunakan angka romawi kecil sebagai penanda list.
- type="I", menggunakan angka romawi besar sebagai penanda list.
- type="1", menggunakan angka desimal sebagai penanda list, ini adalah pilihan default jika atribut type tidak ditulis.

Berikut contoh penggunaan atribut **type** dalam ordered list:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Ordered List</title>
6 </head>
7 <body>
8   <p>Cara membuat telur dadar: </p>
9   <ol type="A">
10    <li>Siapkan daun bawang serta daun seledri, lalu cuci bersih
11    dan potong kecil-kecil.</li>
12    <li>Kocok telur dalam wadah, masukan bumbu halus daun bawang,
```

```
13 seledri dan garam secukupnya.</li>
14 <li>Goreng telur pada wajan, tuangkan adonan pada tengah wajan
15 dan goreng hingga matang.</li>
16 <li>Balik adonan agar sisi atasnya juga ikut matang.</li>
17 <li>Setelah kedua sisi matang, angkat, dan tiriskan.</li>
18 <li>telur dadar siap disajikan.</li>
19 </ol>
20 </body>
21 </html>
```



Gambar: Contoh penggunaan ordered list dengan atribut type="A"

Pada HTML 4.01 atribut `type` berstatus *deprecated*, karena sifatnya yang bersifat tampilan. Tapi di dalam HTML5 atribut ini kembali dinyatakan valid.

Untuk mendapatkan efek yang sama dengan CSS, kita bisa menggunakan property `list-style-type`, seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Ordered List</title>
6 </head>
7 <body>
8   <p>Cara membuat telur dadar: </p>
9   <ol style="list-style-type:decimal-leading-zero">
10    <li>Siapkan daun bawang serta daun seledri, lalu cuci bersih
11    dan potong kecil-kecil.</li>
12    <li>Kocok telur dalam wadah, masukan bumbu halus daun bawang,
13    seledri dan garam secukupnya.</li>
14    <li>Goreng telur pada wajan, tuangkan adonan pada tengah wajan
```

```
15  dan goreng hingga matang.</li>
16  <li>Balik adonan agar sisi atasnya juga ikut matang.</li>
17  <li>Setelah kedua sisi matang, angkat, dan tiriskan.</li>
18  <li>telur dadar siap disajikan.</li>
19 </ol>
20 </body>
21 </html>
```

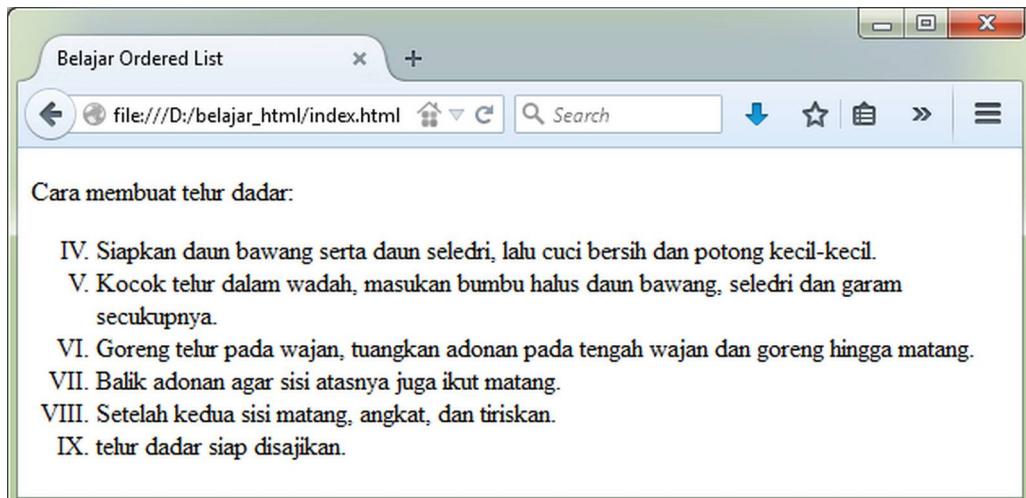
Dengan CSS, jenis karakter yang dihasilkan lebih beragam, bahkan tersedia untuk karakter non-latin seperti bahasa arab atau jepang.

Atribut Start

Atribut **start** digunakan jika kita ingin list dimulai bukan dari angka 1. Nilai dari atribut ini adalah nomor urutan yang diinginkan. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Ordered List</title>
6 </head>
7 <body>
8   <p>Cara membuat telur dadar: </p>
9   <ol type="I" start="4">
10    <li>Siapkan daun bawang serta daun seledri, lalu cuci bersih
11      dan potong kecil-kecil.</li>
12    <li>Kocok telur dalam wadah, masukan bumbu halus daun bawang,
13      seledri dan garam secukupnya.</li>
14    <li>Goreng telur pada wajan, tuangkan adonan pada tengah wajan
15      dan goreng hingga matang.</li>
16    <li>Balik adonan agar sisi atasnya juga ikut matang.</li>
17    <li>Setelah kedua sisi matang, angkat, dan tiriskan.</li>
18    <li>telur dadar siap disajikan.</li>
19 </ol>
20 </body>
21 </html>
```



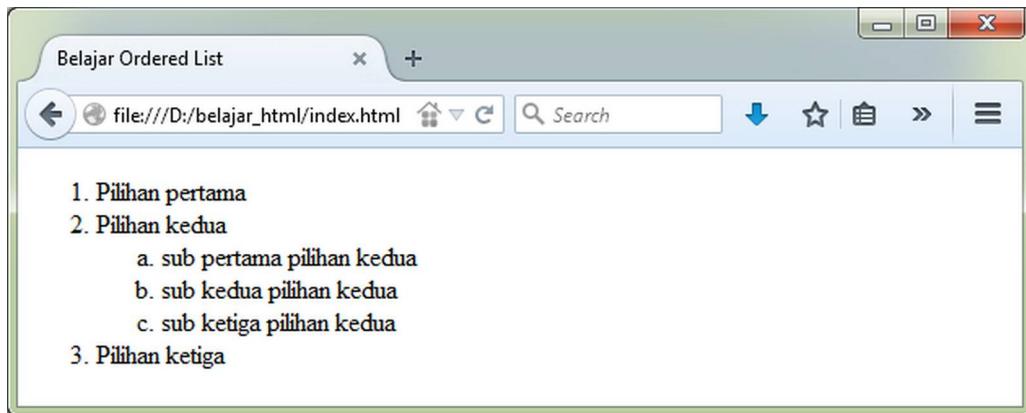
Gambar: Contoh penggunaan ordered list dengan atribut start="4"

Membuat Nested List

Kadang kita juga perlu membuat list yang juga memiliki list di dalamnya, atau dikenal dengan istilah **nested list**. Untuk membuat nested list, caranya dengan menginput tag sebagai bagian dari tag dari list pertama. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Nested Ordered List</title>
6 </head>
7 <body>
8 <ol>
9   <li>Pilihan pertama</li>
10  <li>Pilihan kedua      <!-- tag penutup </li> tidak diletakkan disini -->
11    <ol type="a">
12      <li>sub pertama pilihan kedua</li>
13      <li>sub kedua pilihan kedua</li>
14      <li>sub ketiga pilihan kedua</li>
15    </ol>
16  </li>          <!-- ini adalah tag penutup </li> -->
17  <li>Pilihan ketiga</li>
18 </ol>
19 </body>
20 </html>
```



Gambar: Contoh penggunaan nested ordered list

Untuk membuat nested list, kita harus berhati-hati dengan tempat peletakan penutup tag .

7.2 Unordered List

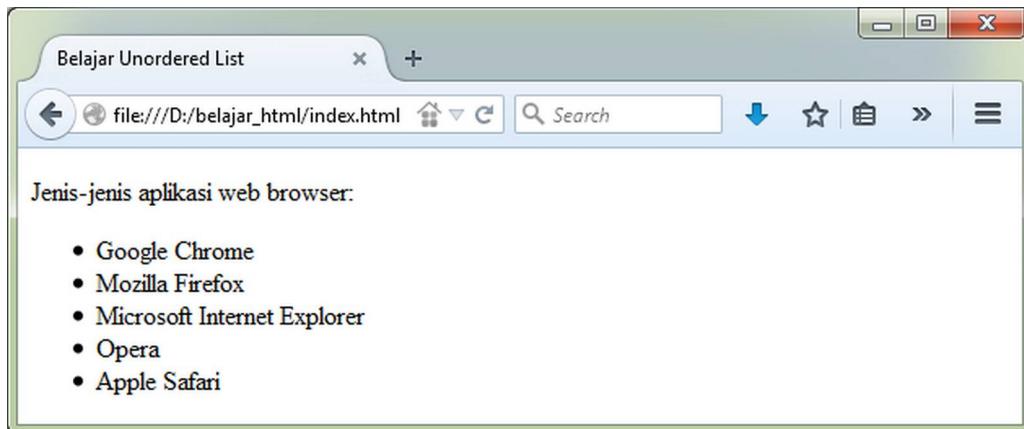
Sesuai dengan namanya, **unordered list** adalah list yang tidak memerlukan urutan. Biasanya list ini digunakan jika posisi daftar tidak berpengaruh.

Di dalam HTML, untuk membuat unordered list kita menggunakan tag . Cara penggunaannya sangat mirip dengan *ordered list*.

Berikut contoh penggunaan unordered list:

index.html

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Belajar Unordered List</title>
6  </head>
7  <body>
8      <p> Jenis-jenis aplikasi web browser: </p>
9      <ul>
10         <li>Google Chrome</li>
11         <li>Mozilla Firefox</li>
12         <li>Microsoft Internet Explorer</li>
13         <li>Opera</li>
14         <li>Apple Safari</li>
15     <ul>
16 </body>
17 </html>
```



Gambar: Contoh penggunaan unordered list

Sama seperti tag , tag juga memiliki atribut **type** untuk mengatur tampilan list.

Atribut Type

Secara default bawaan browser, unordered list biasanya ditampilkan dengan menggunakan lingkaran hitam sebagai penanda list. Dengan menggunakan atribut **type**, kita bisa mengganti karakter ini dengan beberapa karakter lain yang tersedia. Nilai dari atribut type untuk unordered list adalah sebagai berikut:

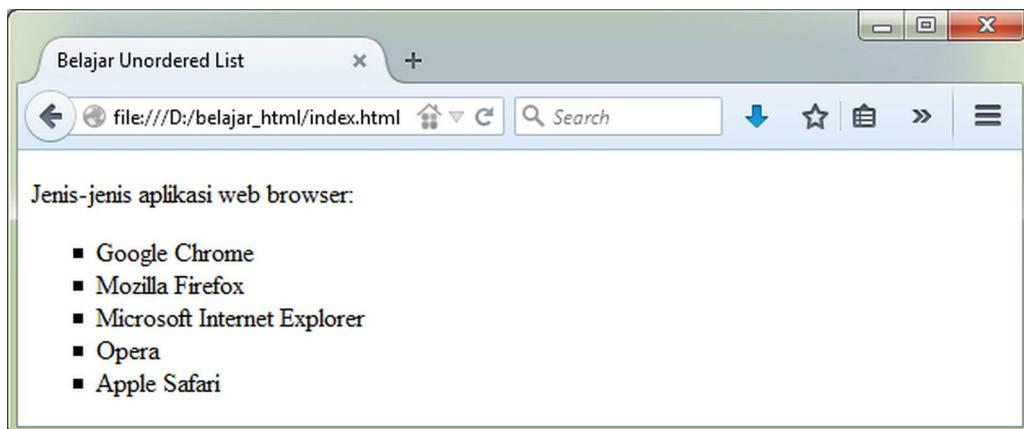
- **type="disc"**, menggunakan bulatan hitam sebagai penanda list, ini adalah pilihan default jika atribut type tidak ditulis.
- **type="circle"**, menggunakan bulatan hitam (dengan isi lingkaran putih) sebagai penanda list.
- **type="square"**, menggunakan kotak hitam sebagai penanda list.

Berikut adalah contoh penggunaan atribut type untuk unordered list:

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Unordered List</title>
6   </head>
7   <body>
8     <p> Jenis-jenis aplikasi web browser: </p>
9     <ul type="square">
10       <li>Google Chrome</li>
11       <li>Mozilla Firefox</li>
12       <li>Microsoft Internet Explorer</li>
13       <li>Opera</li>
14       <li>Apple Safari</li>
```

```
15 <ul>  
16 </body>  
17 </html>
```



Gambar: Contoh penggunaan unordered list dengan atribut type

Berbeda dengan atribut type untuk ordered list, penggunaan atribut type untuk unordered list tidak lagi disarankan (*deprecated*), bahkan di HTML5.

Untuk mendapatkan efek yang sama, kita bisa menggunakan property CSS: **list-style-type**, seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>  
2 <html>  
3   <head>  
4     <meta charset="UTF-8">  
5     <title>Belajar Unordered List</title>  
6   </head>  
7   <body>  
8     <p> Jenis-jenis aplikasi web browser: </p>  
9     <ul style="list-style-type:circle">  
10       <li>Google Chrome</li>  
11       <li>Mozilla Firefox</li>  
12       <li>Microsoft Internet Explorer</li>  
13       <li>Opera</li>  
14       <li>Apple Safari</li>  
15     <ul>  
16   </body>  
17 </html>
```

Selain menggunakan karakter yang disediakan HTML, kita juga bisa mengganti karakter tersebut dengan gambar, atau bahkan tidak menggunakan karakter sama sekali. Metode ini sering digunakan untuk membuat struktur seperti menu navigasi.

Sama seperti ordered list, kita juga bisa membuat struktur nested list dengan unordered list.

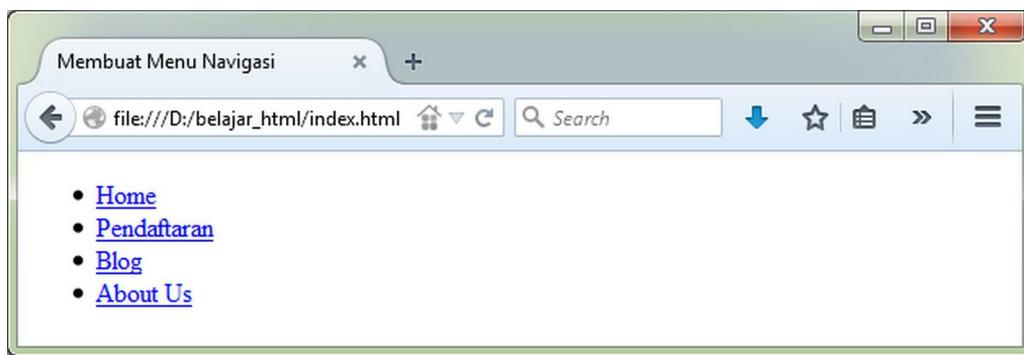
Membuat menu navigasi

Sebagai pelengkap pembahasan mengenai list, saya akan memperlihatkan cara penggunaan unordered list untuk membuat menu navigasi yang umumnya ada di sebuah situs. Menu navigasi adalah menu yang umumnya terdiri dari *home*, *artikel*, *blog*, *contact us*, *about us*, dll.

Berikut adalah contoh pembuatan struktur menu navigasi yang sering digunakan:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Membuat Menu Navigasi</title>
6 </head>
7 <body>
8   <ul>
9     <li><a href="home.html">Home</a></li>
10    <li><a href="pendaftaran.html">Pendaftaran</a></li>
11    <li><a href="blog.html">Blog</a></li>
12    <li><a href="about_us.html">About Us</a></li>
13  </ul>
14 </body>
15 </html>
```



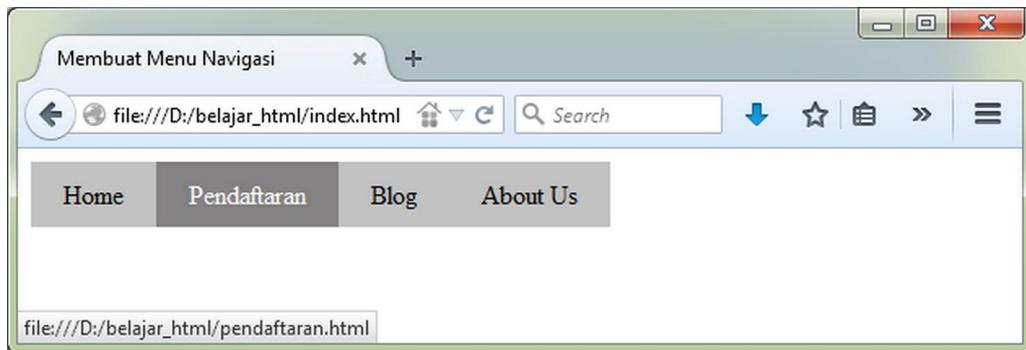
Gambar: Membuat menu navigasi dengan ordered list

Perhatikan bahwa saya menyisipkan link sebagai menu dari list diatas.

Jika anda bertanya kenapa tampilannya tidak seperti menu yang umum ada di internet, ini karena kita belum mengubah efek tampilan dari list tersebut. Dengan sedikit kode CSS, list tersebut bisa 'disulap' menjadi menu navigasi yang 'memukau', seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Membuat Menu Navigasi</title>
6   <style>
7     ul {
8       list-style: none;
9       margin: 0;
10      padding: 0;
11    }
12    ul li {
13      float: left;
14    }
15    ul a {
16      text-decoration: none;
17      color: black;
18      padding: 10px 20px;
19      height: 20px;
20      display: block;
21      background-color: #C2C2C2;
22    }
23    ul a:hover {
24      color: white;
25      background-color: #838181;
26    }
27  </style>
28 </head>
29 <body>
30   <ul>
31     <li><a href="home.html">Home</a></li>
32     <li><a href="pendaftaran.html">Pendaftaran</a></li>
33     <li><a href="blog.html">Blog</a></li>
34     <li><a href="about_us.html">About Us</a></li>
35   </ul>
36 </body>
37 </html>
```



Gambar: Membuat menu navigasi dengan ordered list + CSS

Dapat anda perhatikan bahwa kode HTML untuk kedua tampilan menu ini persis sama. Kita hanya mengubah *style*-nya dari CSS. Dengan struktur HTML yang solid, menu seperti ini bisa dengan mudah dibuat dan diubah-ubah sesuai keinginan (tanpa mengubah kode HTML-nya).

7.3 Description List

Jenis list ketiga yang tersedia di dalam HTML adalah **description list**. Jenis list ini digunakan untuk struktur list seperti ‘*kamus*’, yakni list yang memiliki judul beserta keterangannya. Jenis list ini tidak terlalu sering digunakan.

Untuk membuat description list, seluruh list harus berada di dalam pasangan tag `<dl>` dan `</dl>`. Untuk setiap deskripsi atau bagian judul, kita menggunakan tag `<dt>`. Sedangkan untuk bagian penjelasannya, kita menggunakan tag `<dd>`.

Agar lebih mudah diingat, tag `<dl>` merupakan singkatan dari ‘*description list*’, tag `<dt>` singkatan dari ‘*data term*’, dan tag `<dd>` adalah singkatan dari ‘*data description*’.

Berikut adalah contoh penggunaan description list:

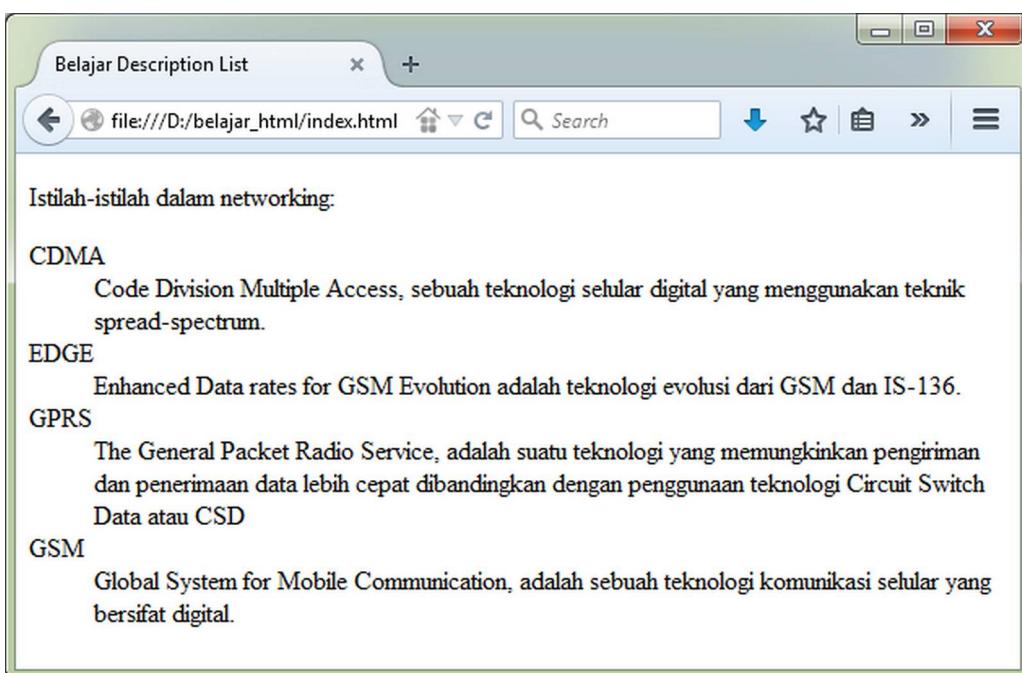
index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Description List</title>
6 </head>
7 <body>
8 <p>Istilah-istilah dalam networking:</p>
9 <dl>
10  <dt>CDMA</dt>
11    <dd>Code Division Multiple Access, sebuah teknologi selular
12      digital yang menggunakan teknik spread-spectrum.</dd>
13  <dt>EDGE</dt>
14    <dd>Enhanced Data rates for GSM Evolution adalah teknologi
15      evolusi dari GSM dan IS-136.</dd>
16  <dt>GPRS</dt>

```

```
17    <dd>The General Packet Radio Service, adalah suatu teknologi  
18    yang memungkinkan pengiriman dan penerimaan data lebih cepat dibandingkan  
19    dengan penggunaan teknologi Circuit Switch Data atau CSD</dd>  
20    <dt>GSM</dt>  
21        <dd>Global System for Mobile Communication, adalah sebuah teknologi  
22        komunikasi selular yang bersifat digital.</dd>  
23    </dl>  
24    </body>  
25    </html>
```



Gambar: Contoh description List <dl>

Seperti yang terlihat, web browser memformat tampilan dari description list dimana bagian penjelasan akan di-*indent* beberapa spasi pada sisi kiri.

Setelah memahami cara pembuatan list di dalam HTML, dalam bab berikutnya kita akan masuk kepada cara menampilkan gambar di dalam HTML. Selanjutnya: **Image Related Element**.

8. Image Related Element

Gambar atau *image* adalah salah satu media terpenting dalam halaman web. Hampir setiap website memiliki gambar sebagai konten maupun sebagai hiasan untuk mempercantik tampilan website. Logo yang menjadi identitas dari sebuah website biasanya juga dibuat dari gambar.

Dalam bab kali ini kita akan membahas dengan detail cara penggunaan gambar di dalam HTML, termasuk cara membuat image map dan figure element.

8.1 Image Element

Untuk dapat menginput gambar ke dalam halaman web, HTML menyediakan **img** element. Agar bisa menampilkan gambar ke dalam web browser, tag `` membutuhkan beberapa atribut, kita akan membahasnya satu persatu.

Tag `` termasuk ke dalam **inline level element**, sehingga akan mengikuti alur teks dimana tag tersebut ditempatkan. Tag `` juga merupakan **void element**, dan tidak membutuhkan tag penutup ``.

Atribut Src

Atribut **src** (singkatan dari **source**) berfungsi untuk menulis alamat gambar yang akan ditampilkan. Alamat ini bisa berupa **alamat absolut** atau **alamat relative** (mengenai perbedaan penulisan alamat ini telah kita pelajari pada bab 6 ketika membahas tentang tag `<a>`).

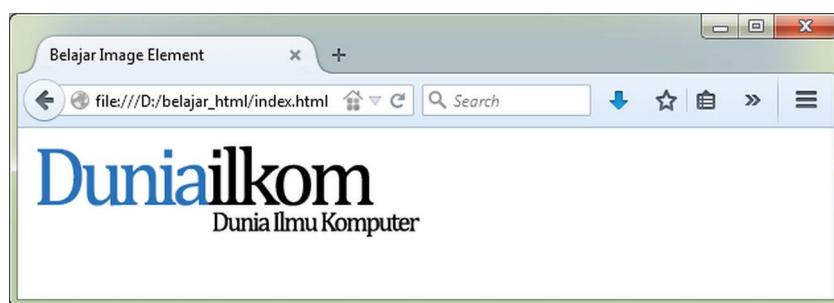


Secara teknis, kita sebenarnya bukanlah ‘menginput gambar’ atau ‘memasukkan gambar’ ke dalam halaman, tetapi membuat ‘link’ ke gambar tersebut. Tetapi untuk mempermudah pemahaman, saya tetap menggunakan istilah ‘menginput gambar’ ke dalam halaman HTML.

Sebagai contoh, kode berikut digunakan untuk menampilkan gambar logo duniaIlkom menggunakan penulisan alamat absolut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   <img src=
9     "http://www.duniailkom.com/wp-content/uploads/2013/09/DuniaIlkomNew.png">
10 </body>
11 </html>
```



Gambar: Contoh menampilkan gambar logo DuniaIlkom dengan alamat absolut

Agar gambar diatas bisa tampil, harus tersedia koneksi internet, karena logo DuniaIlkom tersebut berada di situs www.duniailkom.com.

Untuk dapat menggunakan *alamat relative*, kita harus menyediakan terlebih dahulu gambar yang ingin ditampilkan. Silahkan cari sebuah gambar pada komputer anda dan tempatkan pada folder **belajar_html** (dimana file **index.html** berada).

Sebagai contoh, saya akan menggunakan gambar ‘Bundaran HI’ yang di dapat dari [wikipedia¹](#). Gambar tersebut bernama **Bundaran_HI.jpg**, sehingga tag **** yang digunakan adalah sebagai berikut:

```

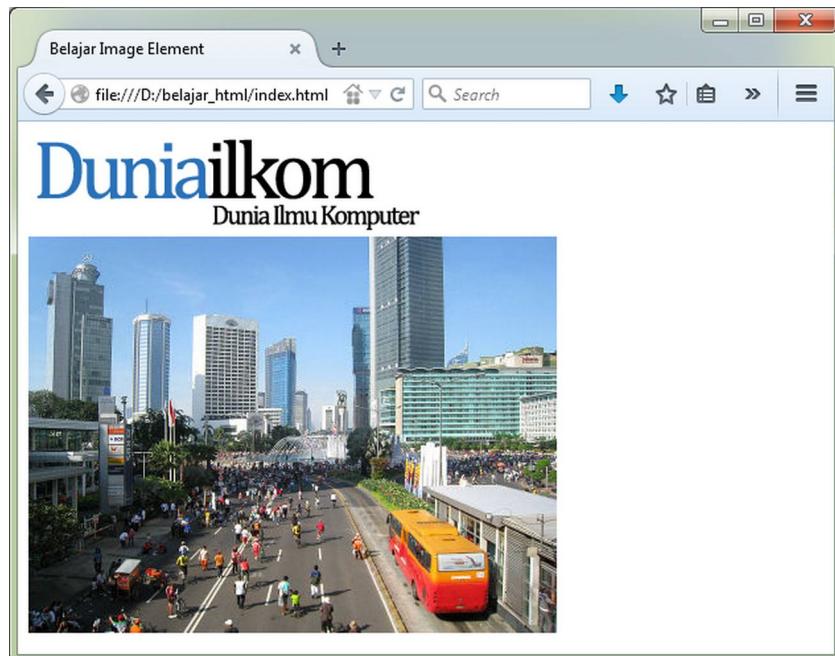
```

Dengan menggunakan template **index.html**, berikut adalah kode HTML cara penggunaan tag **** untuk menampilkan gambar logo DuniaIlkom dan **Bundaran_HI.jpg** :

¹<http://en.wikipedia.org/wiki/Jakarta>

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   <img src=
9     "http://www.duniailkom.com/wp-content/uploads/2013/09/DuniaIlkomNew.png">
10  <br>
11  
12 </body>
13 </html>
```



Gambar: Contoh menampilkan gambar dengan alamat absolut dan relatif

Mengenai format gambar yang didukung oleh HTML, sepenuhnya bergantung kepada web browser. Standar HTML dari W3C tidak secara eksplisit menegaskan tentang format gambar apa saja yang harus digunakan. Namun secara umum, web browser mendukung gambar dengan format *JPG*, *GIF*, *PNG*, *BMP* dan *ICO*.



Hati-hati dengan penulisan atribut *src*. Sering gambar gagal tampil karena kita salah ketik atribut *src* menjadi “*scr*”.

Atribut Alt

Atribut alt (singkatan dari **alternate text**) berfungsi untuk memberikan keterangan mengenai gambar. Isi dari atribut ini adalah pesan teks yang akan ditampilkan pada kondisi-kondisi berikut:

- Gambar tidak dapat ditampilkan (misalkan karena gangguan jaringan internet).
- Web browser tidak mendukung format gambar.
- Web browser sedang dalam proses menampilkan gambar.
- Web browser diset agar tidak menampilkan gambar.
- Pengguna menggunakan browser khusus seperti “*screen reader*” yang tidak akan menampilkan gambar.

Berikut contoh penggunaan atribut alt:

```

```

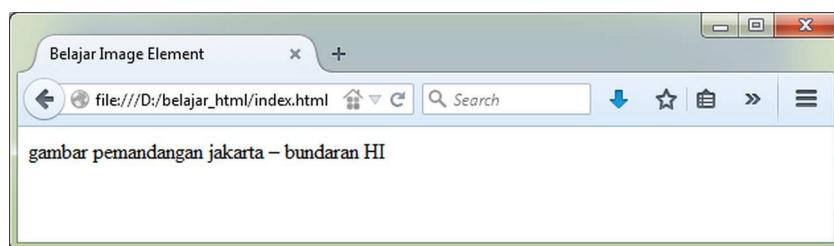
Jika anda menjalankan kode diatas, tidak terlihat perubahan apapun. Agar bisa melihat efek atribut alt, kita harus membuat gambar ‘gagal’ tampil. Sebagai contoh, saya akan mengubah extensi gambar dari **Bundaran_HI.jpg** menjadi **Bundaran_HI.jpgx**, berikut hasil yang didapat:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   
9 </body>
10 </html>

```



Gambar: Teks yang berasal dari atribut alt

Atribut alt juga sangat berguna untuk keperluan SEO. Search engine seperti google tidak dapat memproses gambar dengan maksimal jika ia tidak mengetahui ‘apa yang diceritakan’ dalam gambar tersebut.

Jika kita melakukan pencarian gambar menggunakan google, salah satu hal yang dinilai adalah isi dari atribut alt. Oleh karena itu, gunakan teks deskriptif yang bisa menjelaskan isi gambar dalam atribut ini.

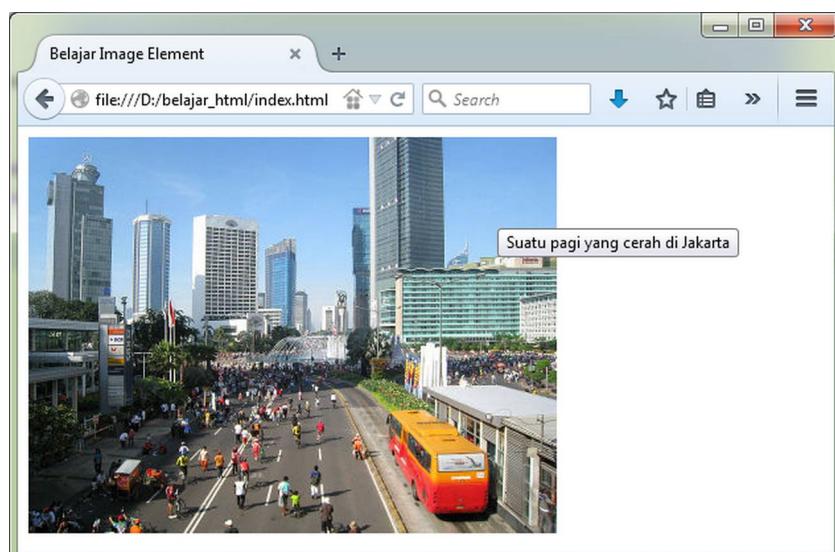
Walaupun sangat bermanfaat, tidak semua gambar harus memiliki atribut alt. Gambar untuk tujuan dekorasi seperti icon atau gambar background tidak perlu menggunakan atribut alt.

Atribut Title

Atribut title bersifat opsional (boleh tidak ditulis), atribut ini digunakan untuk menampilkan keterangan ketika *cursor mouse* berada diatas gambar (*mouse over*). Keterangan ini tampil dalam bentuk tooltip. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   
10 </body>
11 </html>
```



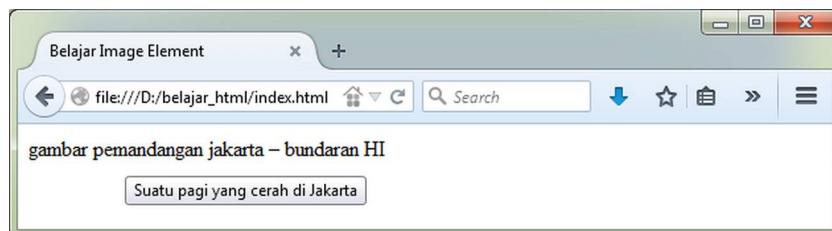
Gambar: Teks title ditampilkan pada saat *mouseover*

Atribut alt dan title berfungsi mirip, tetapi keduanya berbeda. Keterangan di dalam atribut alt akan muncul ketika gambar gagal tampil, sedangkan keterangan di dalam atribut title akan muncul dalam bentuk tooltip dan tidak bergantung apakah gambar tersebut tampil atau tidak.

Apabila gambar gagal tampil, teks keterangan pada atribut title akan tetap ditampilkan, seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   
10 </body>
11 </html>
```



Gambar: Teks title tetap berfungsi walaupun gambar gagal tampil

Atribut Height dan Width

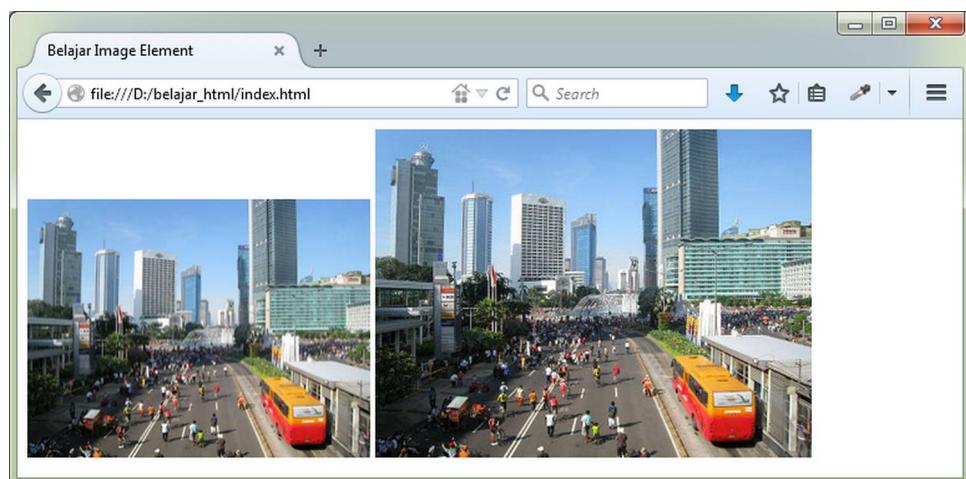
Atribut **height** dan **width** digunakan untuk mengatur ukuran gambar. Atribut *height* untuk mengatur tinggi, sedangkan atribut *width* untuk mengatur lebar gambar. Nilai dari kedua atribut ini berupa angka dalam satuan *pixel* maupun persen (%).

Baik atribut **height** maupun atribut **width** bisa digunakan secara terpisah atau bersamaan. Jika dibuat terpisah, maka web browser akan ‘menyesuaikan’ tinggi atau lebar gambar agar gambar tidak tampil ‘pecah’ (tampil proporsional).

Sebagai contoh, gambar **Bundaran_HI.jpg** memiliki dimensi asli 400×300 pixel (lebar 400 pixel, dan tinggi 300 pixel). Apabila saya menggunakan atribut *width="350"*, web browser secara otomatis menyesuaikan tinggi gambar menjadi **263 pixel**. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   
9   
10 </body>
11 </html>
```



Gambar: Gambar dengan atribut `width="275"` dan `width="350"`

Mengenal Satuan Pixel

Pixel (sering disingkat sebagai px), adalah ukuran standar untuk gambar digital. Dalam teorinya, 1 pixel sama dengan 1 titik terkecil dari layar monitor. Gambar dengan ukuran 200x300px akan ditampilkan di layar dengan lebar 200 titik dan lebar 300 titik.

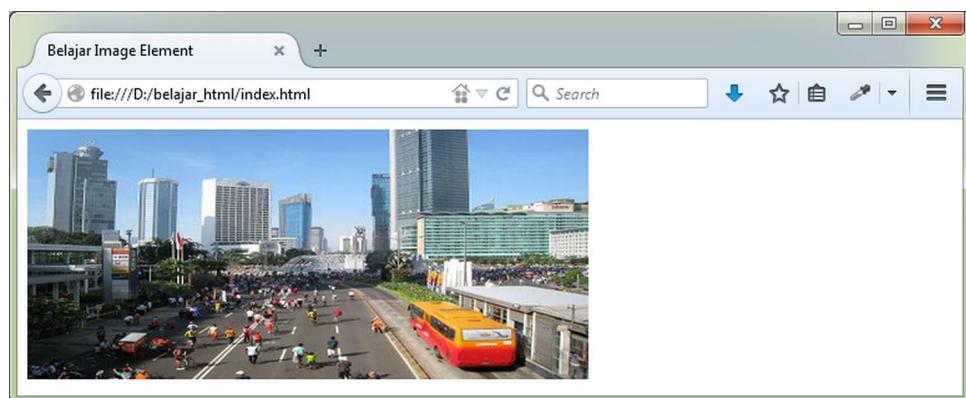
Dalam prakteknya, ukuran pixel ini akan ditampilkan beragam. Smartphone modern sudah menggunakan ukuran pixel yang hampir sama dengan layar komputer, walaupun ukuran layarnya lebih kecil. Pada smartphone ini, gambar dengan ukuran 200x300px bisa jadi ditampilkan dengan 'mengambil' 600x900pixel dari layar smartphone.

Istilah **pixel** berasal dari *picture element*.

Apabila di dalam tag `` dicantumkan atribut `width` dan juga atribut `height`, web browser akan 'memaksa' gambar untuk tampil dalam ukuran tersebut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   
10 </body>
11 </html>
```



Gambar: Gambar ‘dipaksa’ tampil dengan `width="450"` dan `height="200"`

Seperti yang terlihat, gambar sedikit lebih ‘lebar’ karena dipaksa tampil dengan ukuran baru.



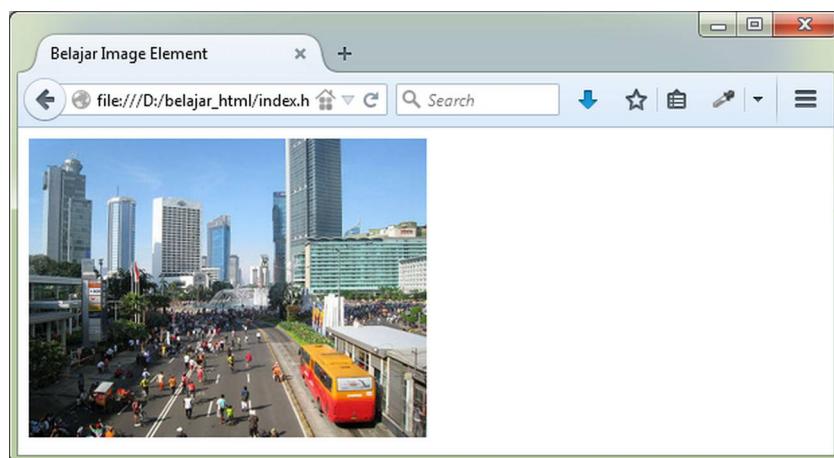
Mengubah lebar dan tinggi gambar dengan atribut `width` atau atribut `height` tidak akan merubah ukuran file (*file size*). Ketika akan menampilkan sebuah gambar, web browser sebenarnya mendownload gambar tersebut terlebih dahulu, baru kemudian menyesuaikan ukuran gambar.

Akibatnya, apabila anda bermaksud menampilkan gambar berukuran kecil, pertimbangkan untuk mengubah ukuran gambar asli daripada memperkecilnya menggunakan atribut `width` atau `height`. Ini dilakukan untuk menghemat penggunaan bandwidth dan mempercepat proses loading halaman web.

Selain mengatur tinggi dan lebar gambar dengan satuan pixel, kita juga bisa menulis ukuran dalam satuan **persen (%)**, seperti berikut ini:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   
10 </body>
11 </html>
```



Gambar: Gambar dengan ukuran relatif lebar 50%

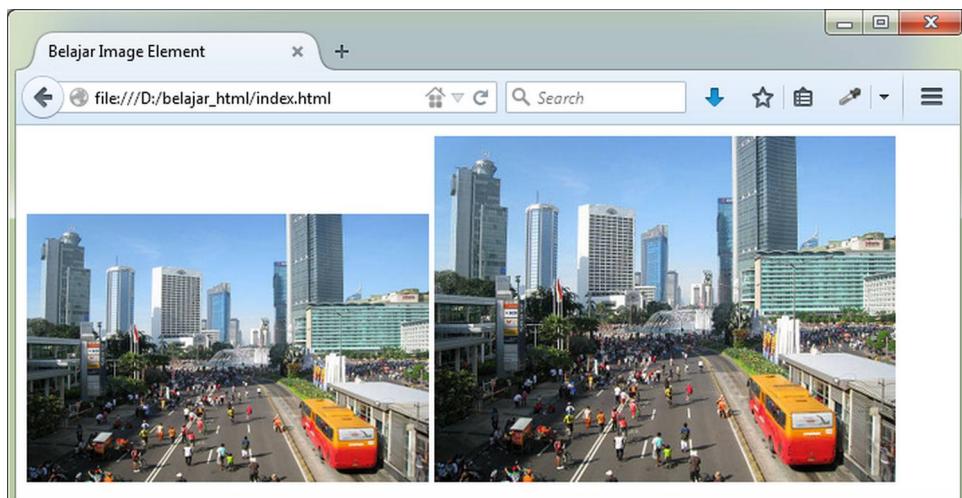
Lebar `width="50%"` berarti ukuran gambar akan diatur sebesar 50% dari lebar *tag parent*. Karena di dalam contoh ini *tag parent* adalah `<body>`, maka lebar gambar akan diatur sebesar 50% dari lebar jendela web browser.

Salah satu hal menarik ketika menggunakan satuan persen adalah, gambar akan mengikuti ukuran lebar jendela web browser. Silahkan anda coba untuk mengecilkan jendela, dan gambar juga akan ikut mengecil. Trik inilah yang digunakan untuk membuat gambar menjadi responsive, yang digunakan di dalam desain **web responsive** (*responsive web design*).

Sebagai alternatif, kita bisa menggunakan CSS untuk mengatur lebar dan tinggi gambar dengan property **width** dan **height**. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   
10  
12 </body>
13 </html>
```



Gambar: Mengatur ukuran gambar dengan CSS

Selain mengatur ukuran gambar, atribut **width** dan **height** juga berguna untuk menghindari efek ‘lompatan’ pada saat loading halaman web. Efek ini terjadi ketika kita mengakses sebuah situs menggunakan koneksi internet yang lambat.

Pada saat halaman web ditampilkan, teks akan diproses terlebih dahulu, baru kemudian gambar ditampilkan satu persatu. Ketika gambar sampai ke web browser, gambar tersebut akan ‘mendorong’ teks di bawahnya, sehingga kita melihat seolah-olah teks ‘melompat’ karena harus menyediakan ruang bagi gambar yang baru saja diterima.

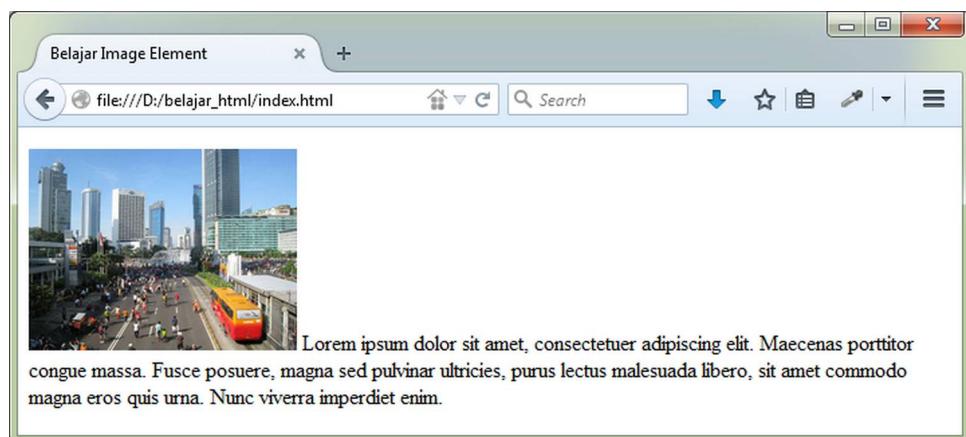
Dengan mengatur ukuran gambar dengan atribut **width** dan/atau **height**, web browser bisa mempersiapkan ‘ruang’ untuk gambar yang belum diterima ini.

Atribut Align

Tag `` merupakan *inline level element*, yang akan mengikuti alur teks dimana tag ini berada. Secara default, gambar akan tampil dengan text berada di sisi bawah gambar, seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   <p>
9     
11    Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas
12    porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies,
13    purus lectus malesuada libero, sit amet commodo magna eros quis urna.
14    Nunc viverra imperdiet enim.
15   </p>
16 </body>
17 </html>
```



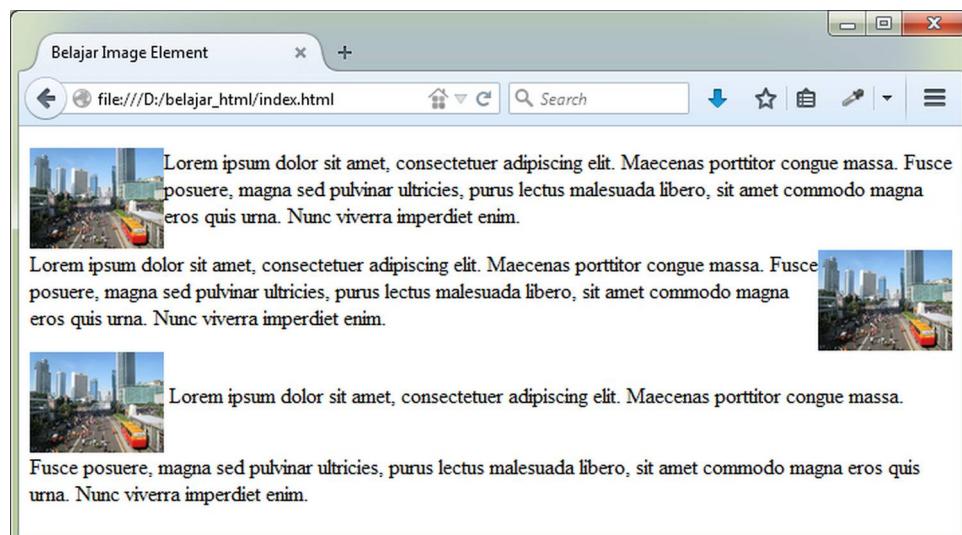
Gambar: Align default web browser

Untuk mengatur posisi tampilan gambar, kita bisa menambahkan atribut **align** ke dalam tag ``. Nilai yang bisa digunakan adalah **bottom**, **left**, **middle**, **right** dan **top**.

Berikut contoh penggunaan atribut **align** dalam tag ``:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   <p>
9     
11     Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas
12     porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies,
13     purus lectus malesuada libero, sit amet commodo magna eros quis urna.
14     Nunc viverra imperdiet enim.
15   </p>
16   <p>
17     
19     Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas
20     porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies,
21     purus lectus malesuada libero, sit amet commodo magna eros quis urna.
22     Nunc viverra imperdiet enim.
23   </p>
24   <p>
25     
27     Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas
28     porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies,
29     purus lectus malesuada libero, sit amet commodo magna eros quis urna.
30     Nunc viverra imperdiet enim.
31   </p>
32 </body>
33 </html>
```



Gambar: Penggunaan atribut align="left", align="right" dan align="middle"

Akan tetapi sebelum mulai menggunakannya, HTML5 menyatakan atribut **align** sudah berstatus **deprecated**. Kita disarankan untuk menggunakan CSS melalui property **float** dan **vertical-align**. Berikut contoh penggunaannya:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   <p>
9     
11     Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas
12       porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies,
13       purus lectus malesuada libero, sit amet commodo magna eros quis urna.
14       Nunc viverra imperdiet enim.
15   </p>
16   <p>
17     
19     Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas
20       porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies,
21       purus lectus malesuada libero, sit amet commodo magna eros quis urna.
22       Nunc viverra imperdiet enim.
23   </p>
24   <p>
25     
27     Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas
28     porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies,
29     purus lectus malesuada libero, sit amet commodo magna eros quis urna.
30     Nunc viverra imperdiet enim.
31   </p>
32 </body>
33 </html>
```

Hasilnya sama persis seperti kode sebelumnya yang menggunakan atribut align.

Apabila anda ingin ada jarak antara gambar dengan teks, bisa menambahkan atribut margin menggunakan CSS:

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Image Element</title>
6   </head>
7   <body>
8     <p>
9       
11      Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas
12      porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies,
13      purus lectus malesuada libero, sit amet commodo magna eros quis urna.
14      Nunc viverra imperdiet enim.
15   </p>
16 </body>
17 </html>
```



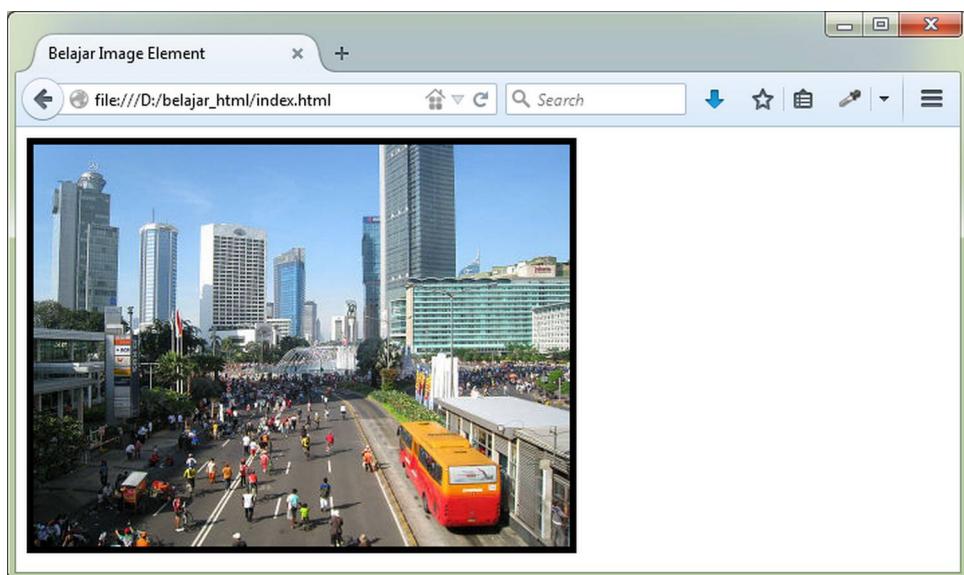
Gambar: Mengatur jarak gambar menggunakan property margin CSS

Atribut Border

Hal yang umum dilakukan untuk mempercantik gambar adalah dengan memberi garis tepi / bingkai. Untuk hal ini kita bisa menggunakan atribut **border**. Atribut border dapat diisi dengan angka dalam satuan *pixel*. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   
10 </body>
11 </html>
```

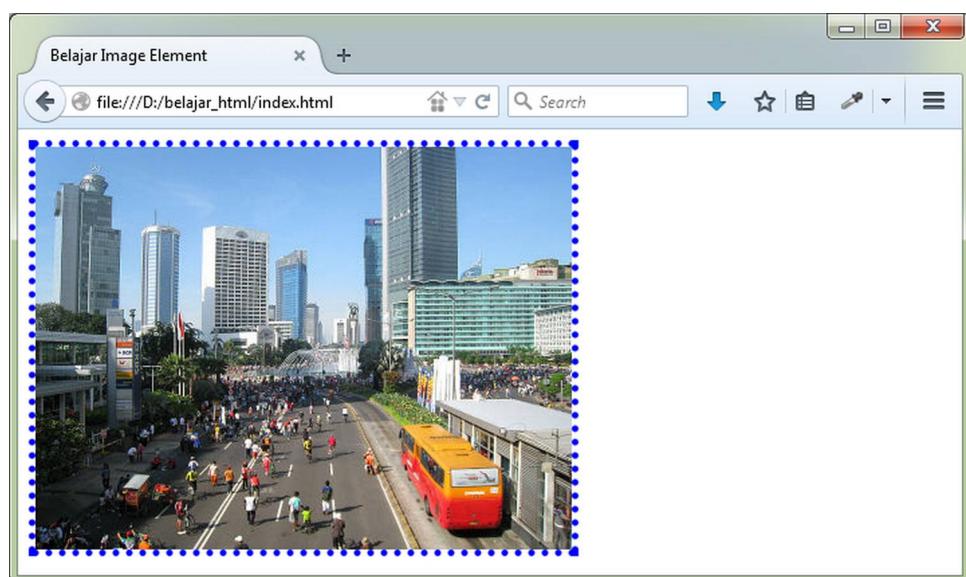


Gambar: Membuat border dengan atribut `border="5"`

Sama seperti atribut **align**, atribut **border** juga telah dinyatakan **deprecated** dalam HTML5. Efek yang sama akan lebih pas jika menggunakan CSS. CSS menyediakan berbagai property untuk membuat border. Berikut salah satu contohnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   
10 </body>
11 </html>
```



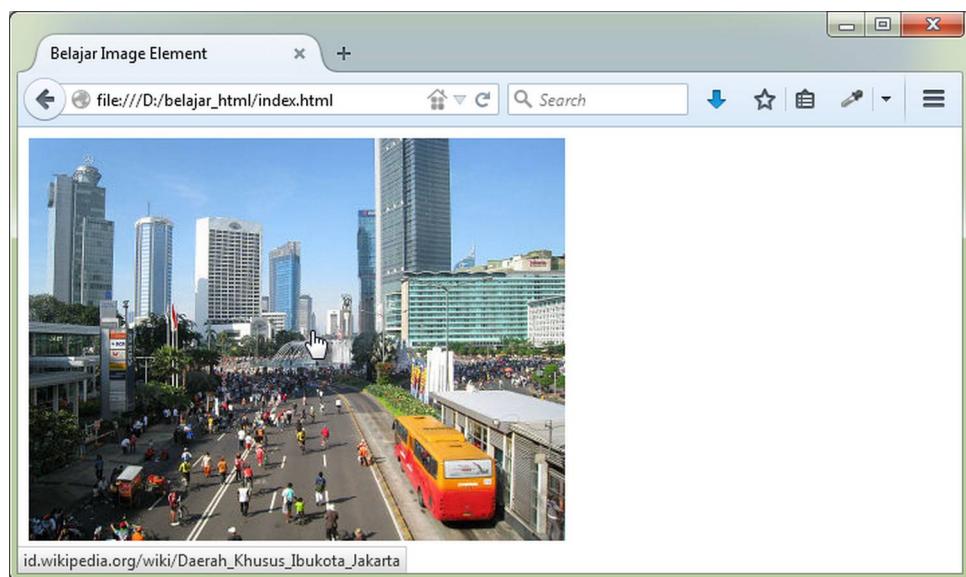
Gambar: Membuat border dengan property **border** CSS

Membuat Link dengan Gambar

Salah satu hal yang sering anda jumpai adalah gambar yang juga berfungsi sebagai **link**. Untuk mendapatkan hasil ini, kita tinggal menyisipkan tag `` di dalam tag `<a>`, seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   <a href="http://id.wikipedia.org/wiki/Daerah_Khusus_Ibukota_Jakarta">
9     
10  </a>
11 </body>
12 </html>
```



Gambar: Gambar sebagai link

Sekarang, apabila anda mengarahkan *cursor mouse* tepat ke atas gambar tersebut, *cursor* akan berubah menjadi tanda ‘*tangan*’ yang menandakan gambar tersebut bisa di-klik. Ketika gambar di klik, web browser akan menampilkan halaman baru sesuai dengan atribut **href** dari tag **<a>**.

Teknik seperti ini juga digunakan untuk membuat gambar *thumbnail*. Yaitu gambar berukuran kecil yang ketika di-klik akan menampilkan gambar versi besar. Untuk membuatnya, cukup menyediakan 2 versi gambar, dan menyesuaikan atribut **href** dari tag **<a>** menuju gambar ukuran besar. Berikut contohnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   <a href="Bundaran_HI_versi_HD.jpg">
9     
10  </a>
11 </body>
12 </html>
```

8.2 Map Element, Area Element, dan Atribut Usemap

Atribut lain yang tersedia dalam tag `` adalah `usemap`. Atribut ini digunakan untuk membuat **image map**, yakni memetakan bagian-bagian gambar untuk dijadikan sebagai link.

Agar bisa menggunakan fitur ini kita membutuhkan bantuan tag `<map>` dan tag `<area>`. Kedua tag ini berfungsi untuk mendefinisikan ‘daerah’ yang akan menjadi link.

Sebagai contoh, saya akan menggunakan gambar **Bundaran_HI.jpg** dan membuatnya menjadi **image map**. Di dalam gambar tersebut terdapat pemandangan bundaran HI, Hotel Indonesia dan Busway. Saya akan membuat ketiga area ini dapat di-klik dan menuju ke halaman Wikipedia yang menjelaskan tentang masing-masing lokasi.

Berikut adalah gambar daerah yang akan menjadi image map:

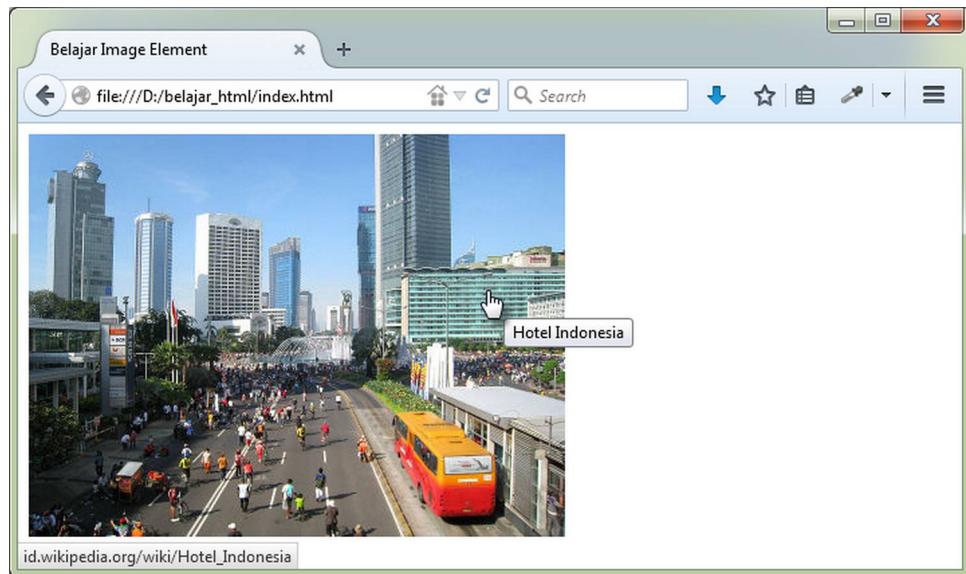


Gambar: Bagian gambar yang akan menjadi imagemap

Dapat anda lihat, saya menandai 3 daerah, yakni **Bundaran HI** dengan lingkaran kuning, **Hotel Indonesia** dengan persegi merah, dan **busway** dengan persegi hijau. Berikut adalah kode HTML untuk membuat image map tersebut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Image Element</title>
6 </head>
7 <body>
8   
10  <map name="map">
11    <area
12      shape="rect" coords="250,175,400,300"
13      href="http://id.wikipedia.org/wiki/Transjakarta"
14      alt="Bus Transjakarta"
15      title="Bus Transjakarta"
16    >
17    <area
18      shape="rect" coords="275,100,400,160"
19      href="http://id.wikipedia.org/wiki/Hotel_Indonesia"
20      alt="Hotel Indonesia"
21      title="Hotel Indonesia"
22    >
23    <area
24      shape="circle" coords="200,150,50"
25      href="http://id.wikipedia.org/wiki/Monumen_Selamat_Datang"
26      alt="Bundaran HI"
27      title="Bundaran HI"
28    >
29  </map>
30 </body>
31 </html>
```



Gambar: Hasil imagemap, setiap area akan menjadi link

Pada baris ke-8 saya menggunakan tag `` seperti biasa, namun kali ini dengan tambahan atribut `usemap="#map"`. Atribut ini berfungsi untuk ‘mengaitkan’ gambar dari tag `` dengan ‘peta gambar’ yang berada pada tag `<map>`.

Tag `<map>` yang berada setelah tag `` memiliki atribut `name="map"`. Nilai atribut ini harus sesuai dengan nilai dari atribut `usemap` pada tag ``. Sebagai contoh, jika dalam tag `` saya mengubah atribut `usemap="#map_punya_saya"`, maka di dalam tag `<map>` juga harus terdapat atribut `name="map_punya_saya"`. Dengan demikian kedua tag ini akan saling terhubung.

Selanjutnya, di dalam tag `<map>` terdapat tag `<area>`. Tag `<area>` digunakan untuk menandai bagian-bagian dari gambar. Karena saya akan membuat 3 daerah, saya menggunakan 3 tag `<area>`.

Pada masing-masing tag `<area>` terdapat atribut **shape**, **coords**, **href**, dan **title**.

Atribut **shape** pada tag `<area>` digunakan untuk membuat bentuk area. Nilai yang bisa digunakan adalah **rect**, **circle**, **poly** dan **default**. Nilai ini berpasangan dengan atribut **coords** yang berfungsi menentukan titik-titik area.

Atribut `shape="rect"` akan membuat area persegi panjang, `shape="circle"` untuk membuat lingkaran, `shape="poly"` untuk membentuk poligon (area dengan bentuk sembarang), dan `shape="default"` akan membuat seluruh bidang gambar menjadi area.

Di dalam HTML, titik koordinat gambar dinyatakan dalam bentuk **x, y**. Titik 0,0 berada pada kiri atas gambar, dan titik 400,300 berada disisi kanan bawah (ini karena saya menggunakan gambar dengan dimensi lebar 400 pixel dan tinggi 300pixel).

Jika menggunakan `shape="rect"`, kita harus menentukan koordinat awal dan koordinat akhir persegi panjang. Dalam contoh diatas, saya menggunakan `shape="rect"` untuk menandai lokasi *busway* dan *Hotel Indonesia*.

Atribut **coords** untuk `shape="rect"` berbentuk **x₁, y₁, x₂, y₂**, dimana **x₁, y₁** menandakan koordinat awal, dan **x₂, y₂** menandakan koordinat akhir. Pada tag `<area>` pertama saya menggunakan `coords="250,175,400,300"` untuk menandai kotak berwarna hijau dimana busway

berada, sedangkan pada tag <area> kedua saya menggunakan coords="275,100,400,160" untuk membuat area Hotel Indonesia.

Dalam tag <area> ketiga, saya menggunakan shape="circle" untuk membuat lingkaran. Kali ini atribut coords akan berbentuk x, y, r. Koordinat x, y adalah titik pusat lingkaran, sedangkan r adalah besar jari-jari lingkaran. Saya menggunakan coords="200,150,50" untuk menandai lokasi bundaran HI.

Atribut shape="poly" menggunakan koordinat dengan bentuk x₁, y₁, x₂, y₂, x₃, y₃, x₄, y₄, ...dst. Atribut ini bisa digunakan untuk membuat area dengan bentuk sembarang, selama koordinat terakhir berimpitan dengan koordinat awal agar area poligon 'tertutup'.

Atribut href dan title pada masing-masing tag <area> digunakan sebagai link dan title untuk area tersebut.

Jika anda menjalankan kode diatas, tiap area yang ditandai akan menjadi link yang masing-masingnya menuju pada halaman terpisah.

Fitur **image map** ini cocok digunakan untuk membuat gambar peta dimana pengunjung bisa langsung berinteraksi dengan gambar.



Salah satu kekurangan image map yang saya gunakan disini adalah HTML tidak memberi 'tanda' untuk bagian gambar yang bisa di-klik. Seluruh gambar tampak 'normal'. Akan lebih baik jika ada sedikit efek ketika mouse berada di atas link area, misalnya mengubah warna gambar, atau menggunakan efek zoom. Hasil seperti ini bisa didapat dengan bantuan JavaScript.

Bagaimana cara mencari titik koordinat gambar?

Untuk mencari titik koordinat gambar, anda bisa menggunakan aplikasi pengolah gambar seperti Photoshop atau Paint bawaan Windows.



Gambar: Mencari koordinat gambar dengan MS Paint

Tampilan diatas adalah dari aplikasi Paint bawaan Windows. Silahkan anda arahkan cursor mouse ke atas gambar. Koordinat cursor mouse tersebut akan ditampilkan di sudut kiri bawah.

Alternatif lain, anda bisa menggunakan online tools *Image Map Generator* seperti: imagemap-generator.dariodomi.de², atau [image-maps.com](https://www.image-maps.com)³

8.3 Figure Element dan Figcaption Element

HTML5 membawa 2 buah **semantic element** baru yang berhubungan dengan gambar, yakni tag `<figure>` dan `<figcaption>`.



Pembahasan tentang **semantic element** akan kita bahas dalam bab khusus tentang **HTML5 semantic element**.

Tag `<figure>` berfungsi sebagai ‘*container*’ bagi satu atau beberapa tag ``. Sedangkan tag `<figcaption>` digunakan untuk membuat keterangan (caption) dari gambar.

Dalam spesifikasi HTML5, tag `<figure>` dapat digunakan untuk gambar, ilustrasi, diagram maupun komponen lain yang berkaitan dengan teks yang sedang dibahas, namun tidak harus berada di satu lokasi yang tetap.

Sebagai contoh, jika saya memiliki artikel yang membahas tentang kota jakarta, gambar tentang *bundaran HI*, atau *bus TransJakarta* tentunya masih berhubungan dengan teks. Akan tetapi gambar tersebut tidak harus berada di posisi tertentu. Gambar bundaran HI bisa ditempatkan diawal artikel, tengah, maupun akhir artikel, dan tidak mempengaruhi fungsi dari gambar tersebut.

Berikut adalah contoh penggunaan tag `<figure>`:

```
<figure>
  
</figure>
```

Dari hasil yang terlihat, web browser akan menambahkan sedikit ‘spasi’ dari tepi tag `<figure>` (dikenal dengan istilah *margin*). Spasi ini nantinya bisa diatur menggunakan CSS.

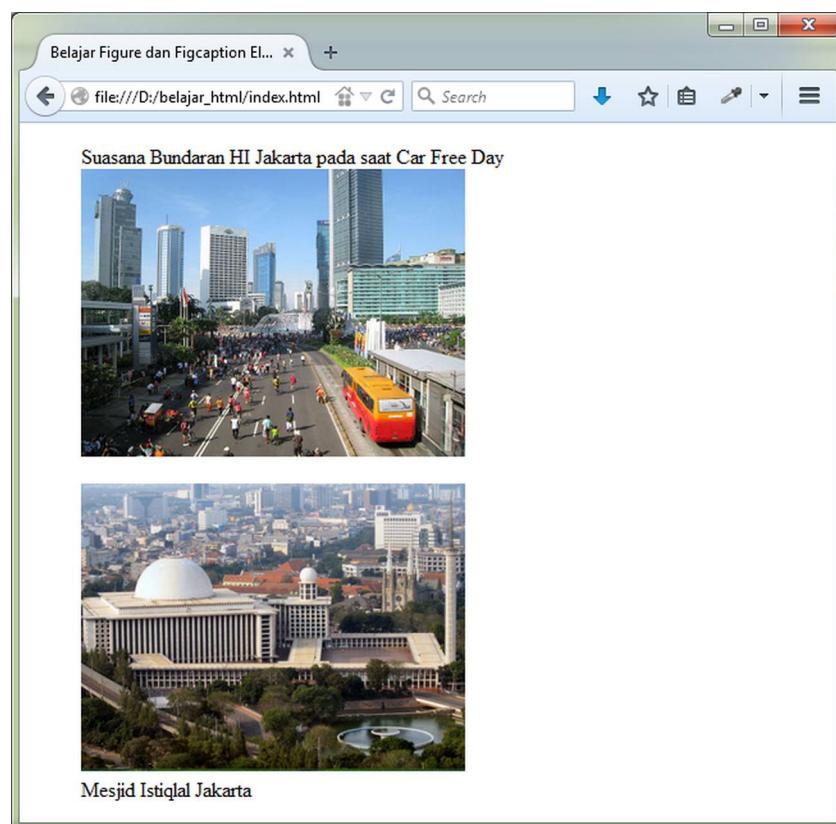
Tag `<figcaption>` bisa ditambahkan untuk membuat keterangan mengenai gambar. Tag ini bisa ditempatkan sebelum maupun setelah ``, seperti contoh berikut:

²<http://imagemap-generator.dariodomi.de>

³<https://www.image-maps.com>

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Figure dan Figcaption Element</title>
6   </head>
7   <body>
8     <figure>
9       <figcaption>
10      Suasana Bundaran HI Jakarta pada saat Car Free Day
11    </figcaption>
12    
13  </figure>
14  <figure>
15    
16    <figcaption> Mesjid Istiqlal Jakarta </figcaption>
17  </figure>
18 </body>
19 </html>
```

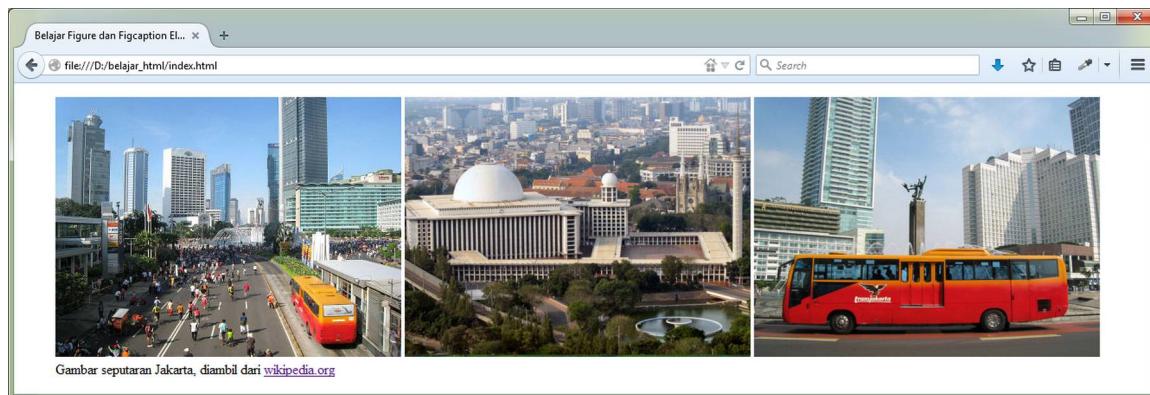


Gambar: **figcaption** bisa ditempatkan sebelum dan sesudah **img**

Spesifikasi HTML5 menyarankan hanya menggunakan 1 tag `<figcaption>` untuk setiap tag `<figure>`. Namun kita bisa menempatkan beberapa gambar di dalam tag `<figure>`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Figure dan Figcaption Element</title>
6   </head>
7   <body>
8     <figure>
9       
10      
11      
12      <figcaption>
13        Gambar seputaran Jakarta, diambil dari
14          <a href="http://en.wikipedia.org/wiki/Jakarta">wikipedia.org</a>
15      </figcaption>
16    </figure>
17  </body>
18 </html>
```



Gambar: Penggunaan beberapa gambar pada tag `<figure>`

Sesuai dengan definisinya, tag `<figure>` tidak hanya berfungsi sebagai ‘penampung’ tag ``, tag ini juga bisa diisi konten lain seperti `<canvas>`, `<svg>`, bahkan berbentuk teks seperti kutipan.

Penutup: Image Related Element

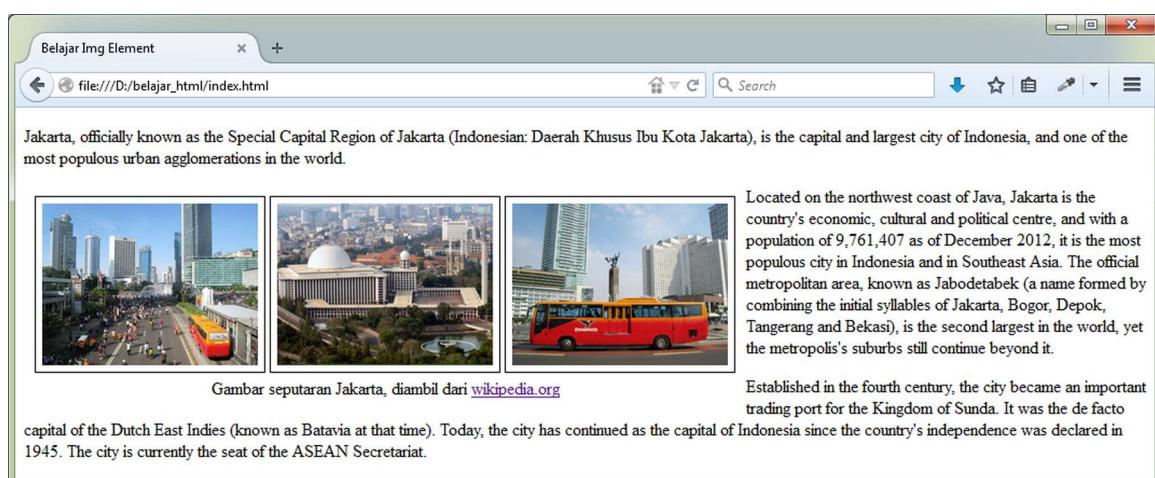
Beberapa tag dan atribut yang kita bahas di dalam bab ini digunakan untuk menampilkan gambar ke dalam web browser. Untuk website modern, hampir mustahil membuat website tanpa gambar, paling kurang gambar digunakan sebagai logo atau sebagai gambar latar.

Berikut adalah contoh kode HTML yang menggunakan berbagai tag dan atribut yang telah kita pelajari dalam bab ini:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Img Element</title>
6   <style>
7     figure{
8       float:left;
9       margin:10px;
10    }
11   img{
12     border: solid 1px black;
13     padding:6px;
14   }
15   figcaption{
16     text-align:center;
17   }
18 </style>
19 </head>
20 <body>
21 <p>Jakarta, officially known as the Special Capital Region of Jakarta
22 (Indonesian: Daerah Khusus Ibu Kota Jakarta),
23 is the capital and largest city of Indonesia,
24 and one of the most populous urban agglomerations in the world.</p>
25
26 <figure>
27   
29   
31   
33   <figcaption>
34     Gambar seputaran Jakarta, diambil dari
35     <a href="http://en.wikipedia.org/wiki/Jakarta">wikipedia.org</a>
36   </figcaption>
37 </figure>
38
39 <p>Located on the northwest coast of Java,
40 Jakarta is the country's economic, cultural and political centre,
41 and with a population of 9,761,407 as of December 2012,
42 it is the most populous city in Indonesia and in Southeast Asia.
43 The official metropolitan area, known as Jabodetabek
44 (a name formed by combining the initial syllables of Jakarta, Bogor,
```

```
45 Depok, Tangerang and Bekasi), is the second largest in the world,  
46 yet the metropolis's suburbs still continue beyond it.</p>  
47  
48 <p>Established in the fourth century, the city became an important  
49 trading port for the Kingdom of Sunda. It was the de facto capital  
50 of the Dutch East Indies (known as Batavia at that time).  
51 Today, the city has continued as the capital of Indonesia since  
52 the country's independence was declared in 1945.  
53 The city is currently the seat of the ASEAN Secretariat.</p>  
54 </body>  
55 </html>
```



Gambar: Artikel tentang Jakarta

Dalam kode diatas, saya menggunakan sedikit CSS untuk memformat tampilan gambar. Anda hanya perlu memahami struktur HTML yang digunakan. Dengan struktur HTML yang baik, halaman web akan mudah diproses oleh web browser, dan juga mudah di-style menggunakan CSS.

Dalam bab ini kita telah membahas berbagai tag dan atribut yang berkaitan dengan gambar. Dalam bab berikutnya, kita akan membahas tentang multimedia, yakni bagaimana cara menampilkan **audio** dan **video** ke dalam halaman HTML.

9. Audio dan Video Element

Pada awalnya, HTML hanya digunakan sebagai sarana berbagi hasil penelitian antar ilmuwan, dimana sebagian besar konten terdiri dari teks, tabel, dan gambar.

Saat ini berbagai situs web saling berlomba menampilkan konten multimedia untuk menarik perhatian pengunjung. Salah satunya dengan menambahkan file multimedia seperti audio dan video.

Sebelum kehadiran HTML5, untuk menjalankan file multimedia, kita harus bergantung kepada aplikasi pihak ketiga seperti **adobe flash**. Oleh karenanya, setiap komputer harus terinstall **adobe flash player**. Jika pengunjung tidak memiliki aplikasi adobe flash player, video tidak bisa dijalankan.

Keterbatasan inilah yang membuat berbagai pihak mendesak agar HTML dapat mendukung audio dan video tanpa bantuan aplikasi lain. **W3C** dan **WHATWG** menjawab masalah ini dengan menyediakan tag `<audio>` dan `<video>`.



Tag `<audio>` dan `<video>` adalah fitur HTML5 yang relatif baru. Dengan demikian, tidak semua web browser mendukung tag ini. Secara umum versi terbaru web browser Google Chrome, Mozilla Firefox, dan Opera mendukung penuh tag `<audio>` dan `<video>`. Yang sering menjadi masalah adalah web browser Internet Explorer 8 (bawaan Windows 7) dan versi IE di bawahnya.

9.1 Audio Element

HTML5 menyediakan tag `<audio>` untuk menjalankan file audio. Berikut adalah cara penggunaan tag `<audio>`:

```
<audio src="lagu.mp3"></audio>
```

Seperti yang terlihat, penulisan tag `<audio>` mirip dengan tag ``, namun tag ini harus ditulis berpasangan dengan tag penutup `</audio>`.

Jika anda menjalankan kode di atas, tidak akan terlihat tampilan apa-apa di dalam web browser. Kita perlu menambahkan beberapa atribut yang akan dibahas setelah ini.

9.2 Atribut Src

Atribut `src` (*source*) digunakan sebagai penunjuk lokasi file audio. Dalam contoh di atas, file `lagu.mp3` harus berada dalam folder yang sama dengan file HTML. Anda bisa meletakkan file audio dimana saja selama bisa diakses menggunakan alamat relatif maupun alamat absolut (mengenai perbedaan kedua alamat ini telah kita bahas pada bab 6).

9.3 Atribut Controls

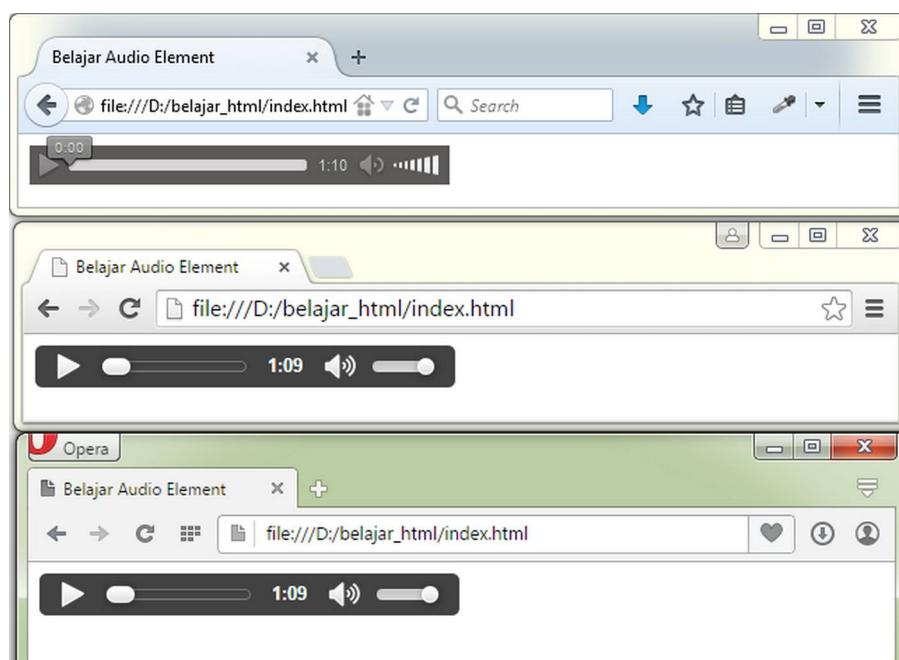
Agar kita bisa memutar file audio, perlu ditambahkan atribut **controls** kepada tag `<audio>`, seperti berikut ini:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Audio Element</title>
6 </head>
7 <body>
8   <audio src="lagu.mp3" controls></audio>
9 </body>
10 </html>
```

Atribut **controls** berfungsi untuk menampilkan tombol *control* pada audio player, yakni tombol play/stop, slider, volume, timer, dll. Jika tag `<audio>` ditulis tanpa menggunakan atribut **controls**, tidak akan terlihat tampilan apa-apa.

Tampilan player untuk tag `<audio>` berbeda dari browser ke browser. Anda dapat melihat perbedaan tampilan audio player pada berbagai web browser dari gambar di bawah ini:



Gambar: Tampilan audio player pada Firefox (atas), Chrome (tengah), dan Opera (bawah)

9.4 Atribut Autoplay

Kita juga bisa menambahkan atribut **autoplay** pada tag `<audio>`. Atribut ini akan membuat audio langsung berjalan ketika halaman web ditampilkan. Tanpa atribut ini, file audio hanya akan berjalan ketika tombol player di tekan. Berikut contoh penggunaannya:

```
<audio src="lagu.mp3" controls autoplay></audio>
```

Silahkan jalankan kode di atas, dan audio akan langsung terdengar ketika halaman web ditampilkan.

Jika atribut **autoplay** digunakan tanpa atribut **controls**, maka anda dapat membuat halaman web dengan audio yang langsung berjalan, namun tidak bisa dihentikan. Sehingga satu-satunya cara untuk menghentikan musik adalah dengan menutup halaman web. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Audio Element</title>
6 </head>
7 <body>
8   <audio src="lagu.mp3" autoplay></audio>
9 </body>
10 </html>
```

9.5 Atribut Loop

Secara default, audio akan berhenti setelah diputar. Apabila anda ingin audio tersebut otomatis diulang kembali dari awal, bisa menambahkan atribut **loop**. Berikut contohnya:

```
<audio src="lagu.mp3" loop controls></audio>
```

Untuk mencoba efek atribut **loop**, silahkan lompati musik hingga di akhir durasi, dan file audio akan otomatis diputar kembali dari awal. Efek seperti ini cocok untuk membuat musik background yang diputar terus menerus.

9.6 Atribut Muted

Agar efek audio tidak mengganggu, kita bisa menjalankan audio dengan volume=0. Jika pengguna ingin mendengarkan musik, ia tinggal memperbesar volume dari tombol audio player. Untuk hal ini, HTML menyediakan atribut **muted**. Berikut contoh penggunaannya:

```
<audio src="lagu.mp3" controls autoplay muted></audio>
```

Kode di atas akan menjalankan file audio ketika halaman web di jalankan, namun tanpa suara. Jika anda melakukan hal ini, paling tidak informasikan kepada pengguna bahwa sebenarnya terdapat file audio yang di-mute-kan.

9.7 Atribut Preload

Tag `<audio>` menyediakan atribut **preload** yang digunakan untuk mengatur cara web browser *mendownload* file audio ketika halaman web dijalankan. Atau bisa juga dipahami dengan cara web browser men-*buffer* file audio tersebut.

Atribut ini bisa diset dengan salah satu nilai berikut: **none**, **metadata**, dan **auto**.

Pengertian Buffering

Jika anda sering mengakses situs video seperti YouTube, kemungkinan besar akan familiar dengan istilah *buffering*. Buffering adalah proses mendownload sebagian kecil durasi video sebelum video tersebut diputar secara terus menerus.

Ketika kita mulai memutar sebuah audio atau video, web browser akan berusaha mendownload bagian video berikutnya (pada saat yang bersamaan ketika kita sedang memutar video). Hal ini dilakukan dengan harapan bagian video tersebut telah tersedia sebelum kita menjalankan bagian itu. Dengan cara ini, video bisa ditampilkan dengan cepat tanpa harus menunggu seluruh file selesai di download dan tanpa efek terputus-putus.

Atribut **preload** berfungsi untuk mengatur proses *buffering* ini.

Jika kita menggunakan atribut `preload="auto"`, maka web browser akan mencoba mendownload (men-*buffer*) seluruh file audio pada saat web ditampilkan pertama kali, terlepas apakah file audio tersebut akan diputar atau tidak.

Hal ini sebaiknya digunakan ketika kita yakin pengguna akan memutar file tersebut. Karena setiap kali halaman web ditampilkan, proses download langsung berjalan (yang juga akan mengurangi kapasitas bandwidth pengguna)



Beberapa web browser menggunakan `preload="auto"` sebagai nilai default.

Sebaliknya, atribut `preload="none"` akan menginstruksikan web browser untuk tidak men-buffer file audio hingga tombol play di-klik. Saat file audio mulai dijalankan (tombol play di-klik), ketika itu lah web browser baru memulai proses buffering.

Metode ini cocok digunakan jika besar kemungkinan file audio tidak akan dijalankan oleh pengguna, atau di dalam halaman terdapat lebih dari 1 file audio. Karena proses buffering tidak akan berjalan sehingga menghemat bandwidth.

Karena cara kerjanya, atribut `preload="none"` ini juga memiliki kelemahan. Web browser tidak dapat menampilkan durasi audio sebelum tombol play di-klik. Hal ini terjadi karena web browser ‘tidak tahu apa-apa’ tentang file audio tersebut (dimana ia harus men-downloadnya terlebih dahulu). Durasi audio baru akan tampil ketika pengguna mulai menjalankan file audio.

Selain itu, pada saat pertama kali file audio dijalankan mungkin akan terdapat jeda beberapa detik ketika web browser memulai proses buffering.

Sebagai jalan tengah, kita bisa menggunakan atribut `preload="meta"`. Atribut ini akan menginstruksikan web browser untuk mendownload sedikit file audio pada saat halaman di tampilkan, hal ini cukup untuk mendapatkan informasi mengenai file tersebut.

Dari ‘sedikit file’ ini, web browser bisa mengetahui dan menampilkan panjang durasi file audio ketika halaman di load. Akan tetapi, proses buffering baru akan berjalan pada saat kita men-klik tombol play.

Atribut `preload` yang kita bahas disini sebenarnya hanya ‘saran’ untuk web browser. Web browser akan mengabaikan nilai atribut ini dalam situasi tertentu, terutama untuk nilai `preload="auto"`.

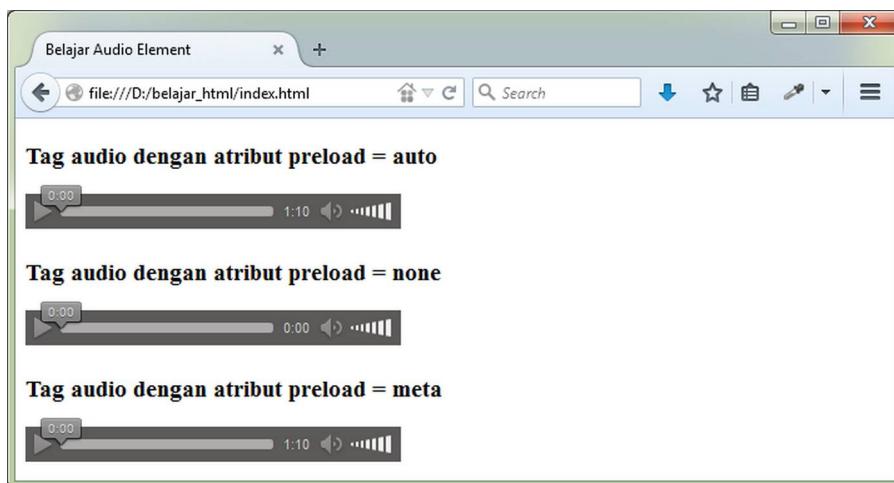
Sebagai contoh, jika web browser mendeteksi pengguna menggunakan koneksi internet yang lambat, maka ia akan beralih ke metoda `preload="meta"` atau `preload="none"`. Beberapa web browser untuk smartphone juga akan mengabaikan nilai `preload="auto"` untuk menghemat bandwidth.

Berikut adalah contoh penggunaan ketiga nilai tersebut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Audio Element</title>
6   </head>
7   <body>
8     <h3>Tag audio dengan atribut preload = auto</h3>
9     <audio src="lagu.mp3" controls preload="auto"></audio>
10    <br>
11    <h3>Tag audio dengan atribut preload = none</h3>
12    <audio src="lagu.mp3" controls preload="none"></audio>
13    <br>
14    <h3>Tag audio dengan atribut preload = meta</h3>
15    <audio src="lagu.mp3" controls preload="meta"></audio>
```

```
16 </body>
17 </html>
```



Gambar: Efek atribut preload pada tag `<audio>`

Jika anda menjalankan kode di atas, tidak akan tampak perbedaan untuk atribut `preload="auto"` dengan `preload="meta"`, hal ini karena efek buffering yang terlalu cepat sehingga tidak terdeteksi (karena file sudah tersedia di komputer kita, maka buffering otomatis tidak diperlukan).

Namun perhatikan tampilan player untuk atribut `preload="none"`. Web browser tidak bisa menampilkan durasi file audio karena belum memiliki informasi mengenai hal ini.

9.8 Mengenal Audio Format

Jika penjelasan mengenai atribut `preload` cukup membuat pusing, maka pembahasan kali ini mungkin akan membuat anda bertambah pusing :)

Walaupun tag `<audio>` merupakan bagian dari element standar pada HTML5, namun HTML5 sendiri tidak memberi penjelasan mengenai format audio apa yang harus didukung oleh web browser.

Terdapat 4 format audio yang umum digunakan di internet, yakni **MP3**, **WAV**, **Ogg**, dan **ACC**.

Format Audio: MP3

Format **MP3** (*MPEG-2 Audio Layer III*) merupakan format paling populer untuk audio. MP3 menggunakan teknik *lossy compression*, dimana kita bisa men-kompres ukuran file audio menjadi lebih kecil dengan mengorbankan sedikit kualitas.

Walaupun populer, format MP3 bersifat *proprietary* (berlisensi). Jika anda ingin mengembangkan aplikasi pemutar MP3 (termasuk membuat web browser yang mendukung MP3) maka harus membayar sejumlah biaya. Karena alasan inilah web browser dari perusahaan ‘kecil’ seperti Opera dan Mozilla Firefox pada awalnya tidak mendukung format MP3. Namun saat ini kedua web browser sudah mendukung penuh MP3 pada versi terbarunya.

Opera menjadi web browser paling akhir yang mendukung MP3, yakni pada versi [Opera 25¹](#) yang dirilis pada September 2014 lalu. Opera versi 24 ke bawah belum mendukung MP3 untuk tag `<audio>`.



Hak cipta MP3 dipegang oleh **Fraunhofer Institute** atau **Fraunhofer Society**, sebuah organisasi penelitian yang berbasis di Jerman. Menurut wikipedia, pada tahun 2005 organisasi ini memperoleh pendapatan sekitar 100 juta euro untuk hak paten format MP3. Penjelasan lebih lanjut bisa anda baca di [Wikipedia/MP3²](#) dan [Wikipedia/Fraunhofer Society³](#).

Format Audio: WAV

Format **WAV** (*Waveform Audio File Format*) juga bersifat *proprietary* (dikembangkan oleh IBM dan Microsoft), namun berbeda dengan **MP3** maupun **Ogg**, WAV menerapkan *lossless compression*, dimana file audio tidak dikompres untuk mempertahankan kualitas. Karena itu file yang dihasilkan berukuran besar sehingga tidak cocok digunakan untuk web.

Format Audio: Ogg Vorbis

Format **Ogg** (atau lengkapnya: *Ogg Vorbis*) adalah format audio *open source*, sehingga penggunaannya tidak membutuhkan biaya lisensi. Ogg menggunakan prinsip *lossy compression* seperti MP3 dimana ukuran file bisa diperkecil dengan mengorbankan sedikit kualitas audio. Format yang dikembangkan oleh **Xiph.Org Foundation** ini pada awalnya dirancang untuk menggantikan MP3. Namun karena relatif baru, penggunaan format ogg masih kalah populer jika dibandingkan mp3.

Karena bersifat *open source*, format Ogg pada awalnya ditetapkan sebagai format audio standar untuk HTML5. Namun hal ini akhirnya dibatalkan dan web browser diberi kebebasan untuk menggunakan jenis format audio. Web browser Mozilla Firefox dan Opera mendukung penuh format Ogg sejak kemunculan tag `<audio>`.

Format Audio: ACC

Format **ACC** (*Advanced Audio Coding*) dikembangkan oleh beberapa perusahaan teknologi seperti **Bell Labs**, **Fraunhofer Institute**, **Dolby Labs**, **Sony** dan **Nokia**. Format *proprietary* ini juga menggunakan *lossy compression* seperti MP3, namun dirancang menghasilkan kualitas suara yang lebih baik dari pada MP3 (dengan ukuran file yang sama). Karena hal inilah format ACC mulai populer digunakan terutama untuk music store seperti **iTunes**.

¹<http://blogs.opera.com/desktop/2014/09/bookmarks-arrive-opera-beta-25>

²<http://en.wikipedia.org/wiki/MP3>

³http://en.wikipedia.org/wiki/Fraunhofer_Society

9.9 Perbedaan Antara Format Audio Dengan File Extension

Format audio yang kita bahas sebelum ini lebih lengkapnya disebut **audio coding format** atau dikenal juga dengan istilah **audio codec** (singkatan dari *coder-decoder*).

Audio codec adalah sekumpulan aturan (algoritma) bagaimana cara mengkonversi/mengkodekan ‘suara’ menjadi data digital yang bisa disimpan di dalam memori komputer. Aturan-aturan inilah yang membedakan sebuah format dengan format lainnya.

Walaupun sebagian besar format audio menggunakan *nama file extension* sesuai dengan singkatan dari codec tersebut, tetapi tidak selalu. Sebagai contoh, format audio **MP3** hanya menggunakan extension **.mp3** sebagai nama file. Akan tetapi format audio **WAV** bisa menggunakan extension **.wav** atau **.wave**.

Format audio **ACC** bahkan memiliki 8 jenis file extension, seperti **.acc**, **.m4a** atau **.mp4** sebagai nama file. Sehingga anda akan menemui sebuah file yang bernama **laguku.m4a**, tetapi format audio yang digunakannya adalah **ACC**. Extension **.m4a** merupakan salah satu format audio yang digunakan oleh **Apple iTunes**.

Tabel di bawah ini merangkum berbagai file extension yang umum digunakan pada sebuah format audio:

Tabel: File extension untuk audio format

Format Audio	File Extension
MP3	.mp3
WAV	.wav, .wave
Ogg	.ogg, .ogv, .oga, .ogx, .ogm, .spx, .opus
ACC	.m4a, .m4b, .m4p, .m4v, .m4r, .3gp, .mp4, .aac

Dari tabel di atas, dapat dilihat bahwa sebuah format audio dapat memiliki banyak *file extension*. Ketika membahas audio dan video digital, file extension ini dikenal juga sebagai ‘*container*’.

9.10 Dukungan Format Audio oleh Web Browser

Banyaknya format audio yang tersedia, didukung beragam oleh web browser. Masalah lisensi umumnya menjadi alasan utama sebuah web browser untuk tidak mendukung format tertentu (seperti kasus MP3 pada Opera).

Tabel di bawah ini merangkum dukungan web browser untuk berbagai format file audio dan mulai versi berapa web browser tersebut bisa memutarinya:

Tabel: Dukungan berbagai format audio pada web browser

	IE	Firefox	Chrome	Opera	Safari (iOS)	Android
MP3	9	22**	5	25	3	2.3
WAV	-	3.6	8	10.5	-	-
Ogg Vorbis	-	3.6	5	10.5	-	-
ACC	9	22**	Yes*	Yes*	3.1	Yes*

* Tidak tersedia data mengenai versi web browser, tetapi saat ini sudah mendukung ACC

**Dukungan format MP3 dan ACC untuk Mozilla Firefox bergantung kepada Sistem Operasi yang digunakan. Sebagai contoh, MP3 dan ACC sudah bisa diputar pada Firefox versi 22 untuk OS Windows, namun baru bisa dijalankan pada Firefox versi 26 untuk OS Linux.

Untuk update mengenai tabel dukungan format audio di atas dapat mengunjungi: [Wikipedia/HTML5_Audio^a](http://en.wikipedia.org/wiki/HTML5_Audio) dan [developer.mozilla.org^b](https://developer.mozilla.org/en-US/docs/Web/HTML/Supported_media_formats)

^ahttp://en.wikipedia.org/wiki/HTML5_Audio

^bhttps://developer.mozilla.org/en-US/docs/Web/HTML/Supported_media_formats

Seperti yang terlihat dari tabel, tidak semua web browser dapat menjalankan file audio dengan format tertentu, dan hanya versi terbaru saja yang mendukung hal tersebut.

Misalnya, jika saya menggunakan file audio dengan format MP3, maka saya tidak bisa memutarnya menggunakan Opera 24 ke bawah (yang masih relatif baru). Namun jika saya menggantinya dengan format Ogg, giliran web browser Internet Explorer yang tidak bisa menjalankannya.

9.11 Menggunakan Beberapa Format File Audio

Salah satu solusi untuk mengatasi keterbatasan dukungan format, kita bisa menggunakan beberapa format untuk file audio yang sama. Web browser kemudian akan memilih format mana yang bisa digunakan.

Sebagai contoh, saya bisa menyediakan 2 buah file audio: Lagu.mp3, dan Lagu.ogg didalam satu tag <audio>. Untuk keperluan ini kita harus menggunakan tag <source>. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Audio Element</title>
6 </head>
7 <body>
8 <audio controls>
9   <source src="Lagu.mp3" type="audio/mp3">
10  <source src="Lagu.ogg" type="audio/ogg">
11 </audio>
12 </body>
13 </html>
```

Kali ini, saya memindahkan atribut `src` dari tag `<audio>` kepada tag `<source>`. Atribut `type` digunakan untuk ‘menginformasikan’ kepada web browser mengenai format file audio. Isi dari atribut ini berupa MIME type file audio (tentang MIME type telah kita bahas pada bab 6).

Ketika kode di atas diakses, web browser bisa memilih menggunakan file format mana yang bisa dijalankan (sesuai urutan penulisan). Google Chrome atau IE akan memilih menggunakan format MP3, sedangkan Opera 24 atau Firefox 20 akan melewatkkan MP3 dan menggunakan format Ogg.

9.12 Pemberitahuan untuk Web Browser tua

Walaupun mayoritas web browser sudah menggunakan automatic update, namun masih terdapat pengguna yang memakai web browser ‘jadul’. Umumnya web browser ini berada di komputer tua atau komputer kantor yang tidak bisa diupdate karena alasan keamanan atau hal lain (misal: IE pada Windows XP hanya bisa diupdate sampai versi 8).

Web browser tersebut tidak bisa menghandle tag `<audio>` karena sama sekali belum memahami HTML5. Metode *polyfill* dengan `HTMLshiv` yang kita bahas pada bab 2 hanya berguna untuk memperkenalkan tag-tag semantic, tetapi tidak bisa ‘mengajari’ web browser bagaimana memainkan audio dan video hanya dengan HTML.

Untuk situasi ini, kita bisa memilih apakah mengabaikan sepenuhnya web browser ini (karena penggunaannya yang memang sudah relatif sedikit) atau memberikan keterangan supaya pengguna mengupdate web browser.

Ketika web browser tidak memahami sebuah tag, maka ia akan melewati tag tersebut. Hal ini bisa kita manfaatkan untuk memberikan informasi terkait hal ini. Berikut contoh penulisan kode HTMLnya:

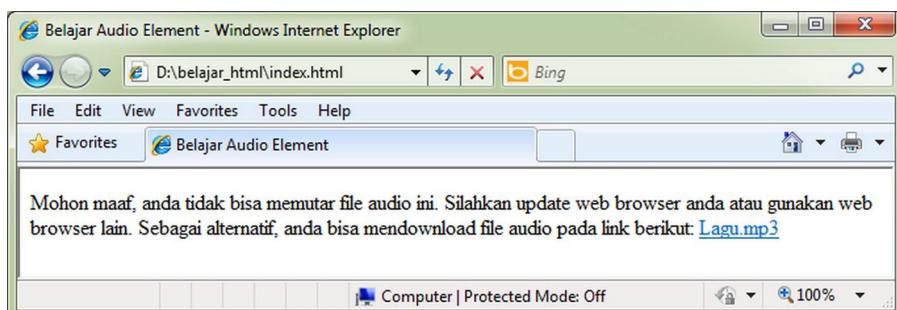
index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Audio Element</title>
6 </head>
7 <body>
8   <audio controls>
9     <source src="Lagu.mp3" type="audio/mp3">
10    <source src="Lagu.ogg" type="audio/ogg">
11    <p>Mohon maaf, anda tidak bisa memutar file audio ini.
12    Silahkan update web browser anda atau gunakan web browser lain.
13    Sebagai alternatif, anda bisa mendownload file audio pada
14    link berikut: <a href="Lagu.mp3" >Lagu.mp3</a></p>
15 </audio>
16 </body>
17 </html>

```

Jika kode di atas dijalankan pada web browser modern, akan tampil audio player bawaan web browser, namun untuk IE 8 yang tidak mendukung HTML5, yang ditampilkan adalah teks yang berada pada tag <p>. Berikut contoh tampilannya pada IE 8 bawaan Windows 7:



Gambar: Tampilan pada IE 8 yang tidak mendukung tag <audio> HTML5

9.13 Video Element

Selain menyediakan fitur pemutar audio, HTML5 juga menyediakan tag <video> untuk menjalankan video. Tag <video> memiliki atribut-atribut yang sama dengan tag <audio>, seperti **src**, **controls**, **autoplay**, **loop**, **muted** dan **preload**.

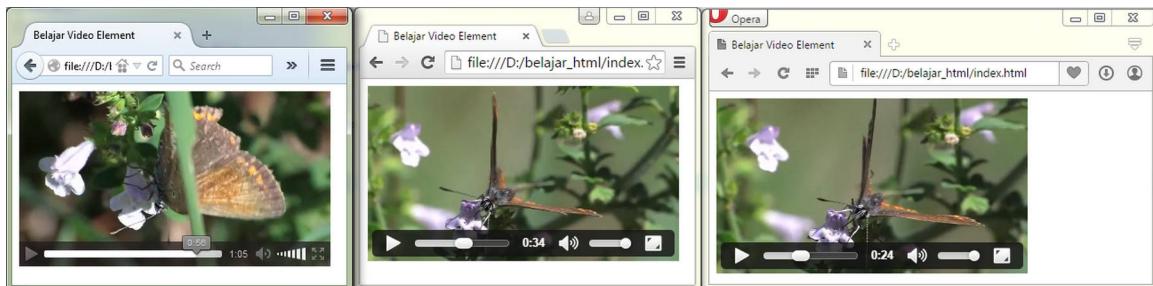


Sebagai file *sample*, saya menggunakan file **butterfly.mp4** yang ditempatkan dalam folder **belajar_html**, sehingga file ini berada pada folder yang sama dengan file **index.html**.

Berikut adalah cara penggunaan tag <video>:

```
<video src="butterfly.mp4" controls></video>
```

Sama seperti tag `<audio>`, atribut `src` digunakan untuk menginput lokasi dan nama file video, sedangkan atribut `controls` ditambahkan agar video player menampilkan tombol control seperti *play/stop, slider, volume* dan *timer*.



Gambar: Tampilan video player pada Firefox (kiri), Chrome (tengah), dan Opera (kanan)

9.14 Atribut Autoplay

Atribut `autoplay` bisa ditambahkan ke dalam tag `<video>` agar video langsung berjalan ketika halaman ditampilkan. Berikut contoh penggunaannya:

```
<video src="butterfly.mp4" autoplay controls></video>
```

9.15 Atribut Loop

Atribut `loop` akan membuat video berulang otomatis ketika sampai di akhir durasi. Tanpa menggunakan atribut ini, video akan berhenti apabila telah selesai diputar. Berikut contoh penulisannya:

```
<video src="butterfly.mp4" loop controls></video>
```

9.16 Atribut Muted

Secara default, video akan dijalankan dengan volume penuh. Kita bisa menambahkan atribut `muted` untuk mematikan volume. Walaupun begitu, pengguna bisa memperbesar volume melalui tombol yang tersedia. Berikut contohnya:

```
<video src="butterfly.mp4" muted controls></video>
```

Menjalankan video tanpa suara bisa digunakan untuk memutar iklan atau promosi produk agar tidak terlalu mengganggu.

9.17 Atribut Preload

Sama seperti tag `<audio>`, tag `<video>` juga memiliki atribut `preload` untuk mengatur cara *buffering* oleh web browser. Nilai yang digunakan pada atribut ini berupa **none**, **metadata**, dan **auto**.

Penjelasan lengkap tentang atribut ini beserta fungsinya telah kita pelajari pada sewaktu membahas tag `<audio>`. Penggunaan atribut ini patut menjadi perhatian jika anda ingin menggunakan nilai `preload="auto"`. Buffering untuk video akan memakan *bandwidth* yang lebih besar dari file audio.

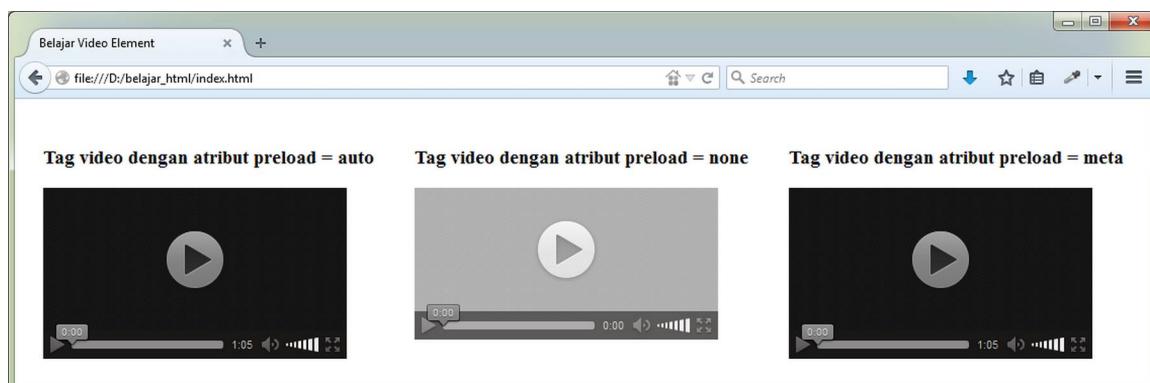
Berikut contoh penggunaan di dalam tag `<video>`:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Video Element</title>
6 </head>
7 <body>
8   <h3>Tag video dengan atribut preload = auto</h3>
9     <video src="butterfly.mp4" controls preload="auto"></video>
10  <br>
11  <h3>Tag video dengan atribut preload = none</h3>
12    <video src="butterfly.mp4" controls preload="none"></video>
13  <br>
14  <h3>Tag video dengan atribut preload = meta</h3>
15    <video src="butterfly.mp4" controls preload="meta"></video>
16 </body>
17 </html>

```



Gambar: Efek atribut `preload` pada tag `<video>`

Seperti yang terlihat, jika kita menggunakan atribut `preload="none"`, web browser tidak dapat menampilkan durasi beserta ukuran tinggi dan lebar jendela video. Hal ini terjadi karena web browser belum mendapat informasi mengenai video tersebut.

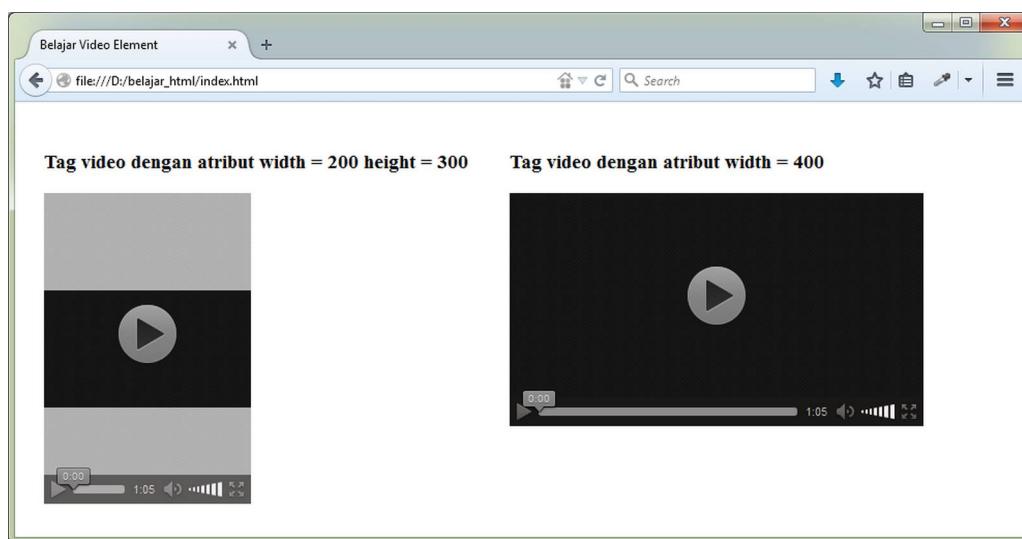
Baru ketika tombol play ditekan, web browser akan menjalankan video dengan ukuran yang seharusnya. Untuk menghindari hal ini kita bisa men-set ukuran video dengan menggunakan atribut **width** dan **height**.

9.18 Atribut Width dan Height

Melalui atribut **width** dan **height**, kita bisa mengatur lebar dan tinggi video yang sedang ditampilkan. Namun berbeda dengan gambar, web browser akan mempertahankan rasio video. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Video Element</title>
6 </head>
7 <body>
8   <h3>Tag video dengan atribut width = 200 height = 300</h3>
9     <video src="butterfly.mp4" controls width="200" height="300"></video>
10  <h3>Tag video dengan atribut width = 400</h3>
11    <video src="butterfly.mp4" controls width="400"></video>
12 </body>
13 </html>
```



Gambar: Mengatur lebar video dengan atribut width dan height

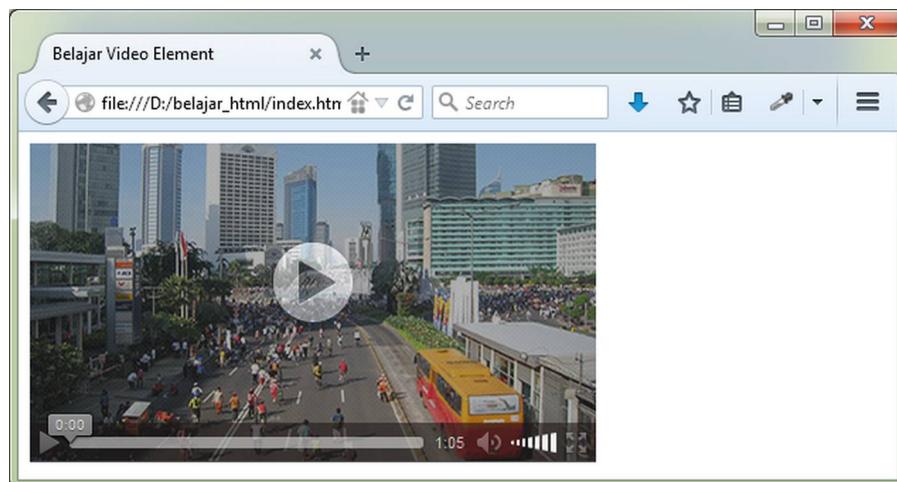
9.19 Atribut Poster

Tag **<video>** memiliki atribut **poster** yang bisa digunakan untuk menampilkan gambar *placeholder* sebelum video dijalankan. Atribut ini berisi nilai alamat kepada file gambar. Berikut

contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Video Element</title>
6 </head>
7 <body>
8   <video src="butterfly.mp4" controls width="400"
9     poster="Bundaran_HI_poster.jpg">
10  </video>
11 </body>
12 </html>
```



Gambar: Cara penggunaan atribut poster untuk video

Ketika halaman web diload, web browser akan menampilkan gambar **Bundaran_HI_poster.jpg** pada jendela video. Dengan menggunakan tag poster ini kita bisa memilih gambar apa yang ingin ditampilkan.

9.20 Mengenal Video Format

Sama seperti file audio, tidak semua format video dapat dijalankan oleh web browser. Sebelum kita melihat keterbatasan dukungan ini, mari kita bahas sejenak tentang format-format file video yang umum digunakan.

Terdapat 3 jenis format video yang cukup populer digunakan, yakni **H.264**, **Ogg Theora** dan **WebM**. Format video dikenal juga sebagai *video codec*.

Format Video: H.264

Format H.264 atau **MPEG-4 Part 10, Advanced Video Coding (MPEG-4 AVC)** merupakan *video codec* yang paling populer digunakan saat ini. Perangkat hardware seperti *video recorder* dan pemutar Blu-ray umumnya menggunakan format ini. Situs video sharing seperti *YouTube* dan *Vimeo* juga menggunakan H.264. Kepopuleran format H.264 mirip dengan format MP3 pada file audio. Format ini dapat menghasilkan video dengan kualitas tinggi namun dengan ukuran file yang relatif kecil.

Walaupun demikian, H.264 termasuk format *proprietary*, sehingga membutuhkan biaya lisensi untuk menggunakannya. Atas dasar hal inilah format H.264 pada awalnya tidak didukung oleh Mozilla Firefox dan Opera.

Mungkin anda tidak terlalu familiar dengan format H.264, namun secara tidak sadar kita telah sering menggunakannya. Format H.264 kebanyakan ditempatkan dalam ‘*container*’ .mp4 atau .mkv. Jika anda menyimpan video dengan extension ini, besar kemungkinan codec yang digunakan adalah H.264. Saya akan kembali membahas tentang ‘*container*’ ini beberapa saat lagi.

Lisensi untuk H.264

Permasalahan lisensi H.264 ternyata tidak hanya untuk pembuat software/hardware. Pembuat video juga bisa terkena dampaknya.

Jika anda membuat video non-komersil menggunakan codec H.264, maka tidak perlu membayar lisensi. Akan tetapi jika anda membuat situs video sharing komersial dengan pengunjung lebih dari 100.000 pengguna per tahun, maka diwajibkan membayar lisensi kepada [MPEG-LA^a](#) selaku pengembang dari codec H.264. Harganya mulai dari US \$25,000 per tahun (Rp 250jt!).

^a<http://www.mpegla.com>

Format Video: Ogg Theora

Jika *Ogg Forbis* merupakan format *open source* untuk file audio, maka *Ogg Theora* adalah format *open source* untuk file video. Codec ini juga dikembangkan oleh perusahaan yang sama, yakni **Xiph.Org Foundation**.

Ketika tahap draft spesifikasi HTML5, Ogg Theora (dan juga Ogg Forbis) sempat diajukan sebagai standar untuk HTML5. Namun karena banyaknya lobi dari beberapa pihak, HTML5 kemudian memilih untuk tidak menetapkan sebuah format tertentu dan memberi kebebasan kepada web browser.

Format **Ogg Theora** didukung oleh komunitas Open Source dan menjadi pilihan format standar oleh Mozilla Firefox dan Opera. Akan tetapi jika dilihat dari hasil kompresi video, dalam beberapa hal hasilnya masih kalah jika dibandingkan dengan H.264.

Format Video: WebM

Format **WebM** berasal dari *video codec VP8* yang dikembangkan oleh perusahaan **On2**. Pada tahun 2010, Google mengakuisisi **On2** dan membuat VP8 sebagai dasar untuk codec *open source* yang dinamakan **WebM**.

Format ini masih relatif baru jika dibandingkan dengan **H.264** dan **Ogg Theora**. Karena didukung oleh perusahaan sebesar Google dan bersifat *open source*, kedepannya **WebM** bisa menjadi format video universal. YouTube juga menyatakan akan beralih ke WebM untuk menggantikan format H.264.

Walaupun begitu banyak kalangan yang masih meragukan format ini, terutama apabila dibandingkan dengan performa format H.264 yang sudah lebih dulu populer.

WebM bisa dijalankan pada Mozilla Firefox, Opera dan Google Chrome. Namun Microsoft memutuskan untuk tidak mendukungnya pada IE.



Selain menggunakan VP8, saat ini Google juga mengembangkan versi yang lebih baru, yakni VP9. Kedua teknologi ini masih termasuk ke dalam **WebM**.

9.21 Mengenal Standar Video

Video codec atau format video yang kita bahas sebelumnya hanya salah satu dari standar yang digunakan untuk membuat video. Setidaknya, terdapat 3 standar di dalam video digital.

Format video seperti **H.264**, **Ogg Theora** dan **WebM** adalah *codec* (singkatan dari *coder-decoder*) yang berisi algoritma mengenai cara menyimpan gambar bergerak (video) ke dalam bit-bit komputer. Ini adalah standar pertama dalam video digital.

Standar kedua yakni mengenai aspek ‘suara’ yang ada di dalam video. Umumnya video digital menggunakan standar terpisah untuk audio, format audio ini sudah kita bahas sebelumnya, dimana MP3, Ogg, Wav dan ACC adalah format audio yang banyak beredar saat ini. Format video **H.264** umumnya menggunakan **MP3** atau **ACC** sebagai *audio codec*, sedangkan **Ogg Theora** dan **WebM** menggunakan **Ogg forbis** untuk pengisi suara.

Standar ketiga adalah ‘**container**’ atau penampung kedua codec di atas (*audio codec* dan *video codec*). Container ini lebih dikenal sebagai file format. Sebagai contoh, file dengan akhiran **.mp4** umumnya digunakan untuk menampung video H.264, akhiran **.ogv** digunakan oleh codec Ogg Theora sedangkan **.webm** digunakan oleh codec **WebM**.

Selain itu, terdapat juga container dengan multi format, seperti **.mkv** yang bisa digunakan oleh **H.264**, **Ogg Theora** maupun **WebM**. Sehingga hanya dengan melihat nama sebuah file, kita belum bisa mendapat informasi mengenai codec yang digunakan. Untuk list lengkap mengenai container ini bisa dilihat di: [Wikipedia/Comparison_of_container_formats⁴](http://en.wikipedia.org/wiki/Comparison_of_container_formats).

⁴http://en.wikipedia.org/wiki/Comparison_of_container_formats

9.22 Dukungan Format Video oleh Web Browser

Masalah lisensi umumnya menjadi faktor terbesar dari penggunaan format H.264. Web browser Mozilla Firefox dan Opera pada awalnya tidak mendukung format ini, namun pada versi terbaru keduanya sudah bisa memutar format H.264. Di lain pihak, Internet Explorer tetap tidak mendukung format Ogg, bahkan pada versi terbarunya.

Tabel berikut merangkum dukungan web browser terhadap format video dalam HTML5:

Tabel: Dukungan berbagai format video oleh web browser

	IE	Firefox	Chrome	Opera	Safari (OS X)	Android
H.264	9	21*	5	24	3.1	3.0
Ogg Theora	-	3.5	5	10.5	-	2.3
WebM	-	4	6	10.6	-	2.3

* Format H.264 didukung beragam oleh Mozilla Firefox dan bergantung kepada sistem operasi. Untuk Windows 7 ke atas, versi 21 sudah bisa menjalankannya, namun untuk OS Linux baru bisa diputar pada versi 34.

Untuk update mengenai tabel di atas dapat mengunjungi: [Wikipedia/HTML5_video^a](http://en.wikipedia.org/wiki/HTML5_video) dan [developer.mozilla.org^b](https://developer.mozilla.org/en-US/docs/Web/HTML/Supported_media_formats).

^ahttp://en.wikipedia.org/wiki/HTML5_video

^bhttps://developer.mozilla.org/en-US/docs/Web/HTML/Supported_media_formats

9.23 Menggunakan Beberapa Format File Video

Sama seperti file *audio*, untuk mengatasi keterbatasan dukungan format *video*, kita bisa menggunakan beberapa jenis format pada 1 tag <video>. Web browser akan memilih format yang bisa diputar sesuai dengan spesifikasinya. Berikut contoh penggunaan berapa format file video dalam 1 penulisan <video>:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Video Element</title>
6 </head>
7 <body>
8 <video controls>
9   <source src="butterfly.mp4" type="video/mp4">
```

```
10 <source src="butterfly.ogg" type="video/ogg">
11 <source src="butterfly.webm" type="video/webm">
12 </video>
13 </body>
14 </html>
```

Dengan menyediakan beberapa format video, bisa mengantisipasi keterbatasan dukungan format pada beberapa web browser.

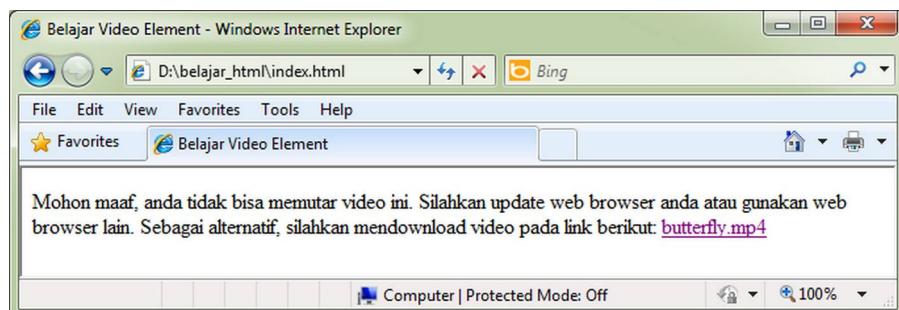
9.24 Pemberitahuan untuk Web Browser Tua

Sama seperti tag `<audio>`, kita juga bisa membuat pesan informasi untuk web browser tua yang belum mendukung HTML5. Berikut adalah contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Video Element</title>
6 </head>
7 <body>
8 <video controls>
9   <source src="butterfly.mp4" type="video/mp4">
10  <source src="butterfly.ogg" type="video/ogg">
11  <source src="butterfly.webm" type="video/webm">
12  <p>Mohon maaf, anda tidak bisa memutar file video ini.
13  Silahkan update web browser anda atau gunakan web browser lain.
14  Sebagai alternatif, anda bisa mendownload file video pada
15  link berikut: <a href="butterfly.mp4">butterfly.mp4</a></p>
16 </video>
17 </body>
18 </html>
```

Jika anda menjalankan kode di atas pada web browser yang mendukung tag `<video>`, maka akan tampil jendela video beserta tombol control. Namun pada web browser yang tidak support HTML5, akan tampil informasi keterangan. Berikut adalah contoh tampilannya pada IE 8:



Gambar: Tampilan pada IE 8 yang tidak mendukung tag <video> HTML5

Dalam bab ini kita telah membahas tentang tag <audio> dan <video> yang dibawa oleh HTML5. Keterbatasan dukungan format memang menjadi kendala tersendiri. Tapi dengan semakin menurunnya popularitas flash player, tidak lama lagi semua web browser akan mencari ‘format bersama’, sehingga kita bisa menggunakan tag <audio> dan <video> secara maksimal.

Dalam bab berikutnya kita akan masuk ke dalam cara pembuatan tabel di HTML.

10. Table Element

Dalam bab ini kita akan membahas tentang cara membuat tabel di dalam HTML. Tabel diperlukan untuk menampilkan data yang tersusun secara sistematis seperti hasil laporan, daftar nilai, dan daftar peserta.

HTML menyediakan beragam atribut dan tag untuk membuat struktur tabel. Akan tetapi sebagaimana yang akan anda pelajari nantinya, sebagian besar atribut ini sudah dinyatakan *deprecated / obsolete* (usang) dalam HTML5. Kita disarankan menggunakan CSS untuk mendapatkan efek yang sama.

Walaupun begitu, bagi pemula memang lebih mudah menggunakan atribut daripada CSS. Sehingga saya tetap membahas cara penggunaan atribut ‘*tua*’ ini, kemudian diikuti dengan *property CSS* yang bisa digunakan untuk menghasilkan efek yang sama.

Sebelum era CSS, tabel juga berfungsi untuk membuat layout/tampilan web seperti *header*, *menu navigasi*, *sidebar* dan *footer*. Saat ini sangat tidak dianjurkan menggunakan tabel untuk membuat struktur web. Namun agar anda menguasai secara penuh perkembangan HTML, di akhir bab ini saya juga menyertakan cara membuat layout menggunakan tabel HTML.



Agar dapat mengikuti penjelasan kode CSS, saya sarankan untuk membaca Appendix: Kursus Kilat CSS di bagian akhir buku ini, terutama jika anda belum pernah bersentuhan dengan CSS.

10.1 Struktur Dasar Tabel (`table`, `td` dan `tr` element)

Untuk membuat tabel, setidaknya diperlukan 3 jenis element, yakni `<table>`, `<tr>`, dan `<td>`. Ketiga tag ini hanyalah syarat minimal. HTML masih menyediakan beragam tag yang bisa digunakan untuk membuat tabel.

Sebelum membuat tabel, sebaiknya kita sudah menentukan berapa jumlah baris dan kolom tabel yang akan dibuat. Berikut adalah kode HTML untuk pembuatan tabel yang terdiri dari 3 baris dan 2 kolom:

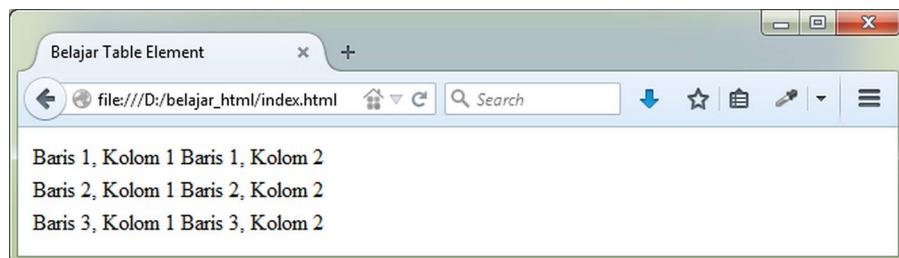
index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Table Element</title>
6 </head>
7 <body>
8 <table>
9   <tr>
10    <td>Baris 1, Kolom 1</td>
11    <td>Baris 1, Kolom 2</td>
12  </tr>
13  <tr>
14    <td>Baris 2, Kolom 1</td>
15    <td>Baris 2, Kolom 2</td>
16  </tr>
17  <tr>
18    <td>Baris 3, Kolom 1</td>
19    <td>Baris 3, Kolom 2</td>
20  </tr>
21 </table>
22 </body>
23 </html>
```

Seperti yang terlihat, seluruh struktur tabel harus berada diantara tag pembuka `<table>` dan tag penutup `</table>`. Tag `<table>` termasuk ke dalam *block level element*, dan ditampilkan terpisah pada baris baru.

Untuk membuat baris pada tabel, kita menggunakan tag `<tr>` (singkatan dari **t**able **r**ow). Selanjutnya, di dalam tag `<tr>`, disisipkan tag `<td>` yang merupakan sel tabel (singkatan dari **t**able **d**ata).

Dalam contoh di atas, saya merancang tabel dengan 3 baris dan 2 kolom, sehingga saya perlu membuat 3 buah tag `<tr>` yang di dalam masing-masingnya terdapat 2 buah tag `<td>`. Berikut adalah tampilannya di dalam web browser:



Gambar: Struktur dasar tabel HTML

Apakah ini tabel?

Seperti yang terlihat, hasilnya tidak tampak seperti tabel. Ini adalah tampilan default bawaan web browser. Web browser akan menampilkan tabel di atas tanpa bingkai (*border*). Oleh karena itu, kita perlu men-set ukuran bingkai tabel.

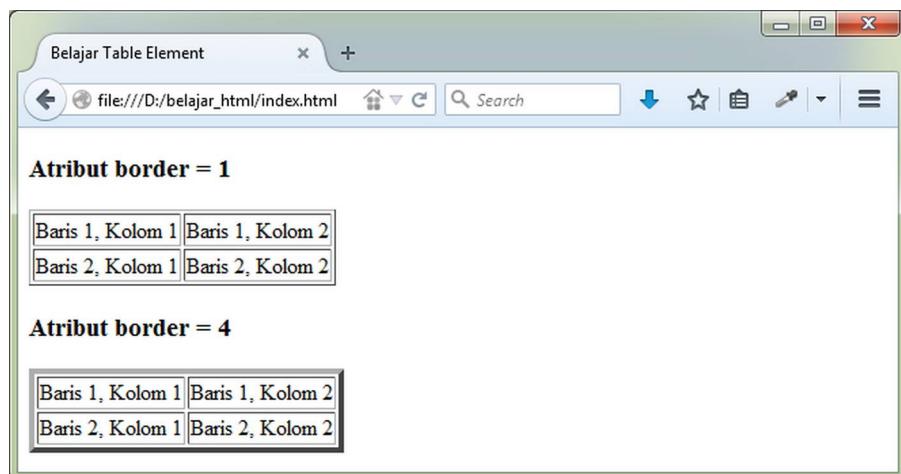
10.2 Atribut Border

Untuk menampilkan bingkai tabel (dikenal dengan istilah *border*), kita bisa menambahkan atribut *border* ke dalam tag `<table>`. Nilai dari atribut ini berupa angka dalam satuan *pixel*. Sebagai contoh, `border="1"` akan menampilkan tabel dengan bingkai sebesar 1 pixel, sedangkan `border="3"` akan menghasilkan tabel dengan bingkai berukuran 3 pixel.

Berikut contoh penggunaan atribut *border* pada tabel:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Table Element</title>
6 </head>
7 <body>
8 <h3>Atribut border = 1</h3>
9 <table border="1">
10  <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
11  <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
12 </table>
13 <h3>Atribut border = 4</h3>
14 <table border="4">
15  <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
16  <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
17 </table>
18 </body>
19 </html>
```



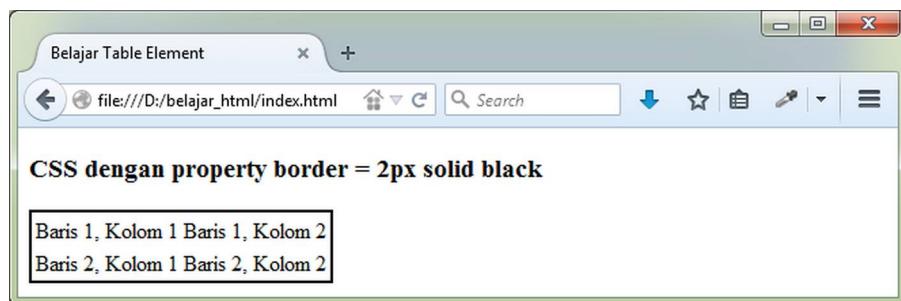
Gambar: Penggunaan atribut border pada tabel HTML

Dari tampilan di atas, perlu dipahami bahwa atribut border digunakan untuk membuat bingkai di sisi luar tabel. Anda bisa melihat efeknya ketika saya menggunakan atribut border="4", dan yang terlihat tebal hanyalah border pada sisi luar tabel. Untuk garis pembatas antar baris / kolom, akan diatur menggunakan atribut lainnya.

Di dalam HTML5, atribut border sudah dianggap *deprecated* dan tidak disarankan untuk digunakan lagi. Sebagai gantinya, kita bisa menggunakan CSS dengan property border. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Table Element</title>
6 </head>
7 <body>
8 <h3>CSS dengan property border = 2px solid black</h3>
9 <table style="border: 2px solid black">
10  <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
11  <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
12 </table>
13 </body>
14 </html>
```

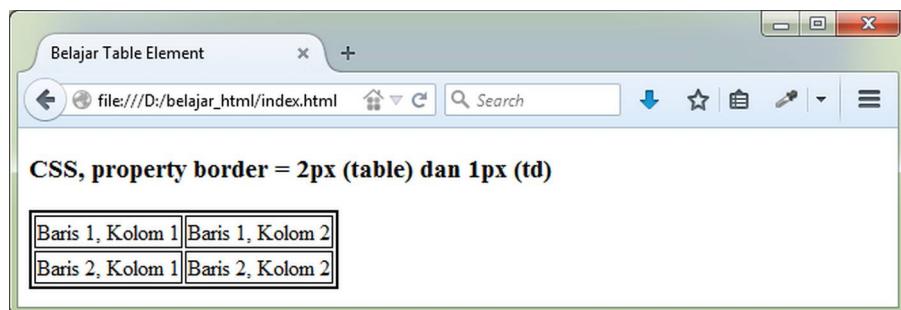


Gambar: Penggunaan property border CSS pada tabel HTML

Kode di atas akan menghasilkan tabel dengan ketebalan *border* 2 pixel dan menggunakan garis normal (*solid*) berwarna hitam (*black*). Jika anda ingin setiap sel tabel juga memiliki border (seperti hasil jika menggunakan atribut border), maka property border harus ditempatkan pada setiap tag <td>, seperti kode berikut ini:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Table Element</title>
6 </head>
7 <body>
8 <h3>CSS, property border = 2px (table) dan 1px (td)</h3>
9 <table style="border: 2px solid black">
10  <tr>
11    <td style="border: 1px solid black;">Baris 1, Kolom 1</td>
12    <td style="border: 1px solid black;">Baris 1, Kolom 2</td>
13  </tr>
14  <tr>
15    <td style="border: 1px solid black;">Baris 2, Kolom 1</td>
16    <td style="border: 1px solid black;">Baris 2, Kolom 2</td>
17  </tr>
18 </table>
19 </body>
20 </html>
```



Gambar: Penggunaan property border CSS pada tag <table> dan <tr>

Jika menggunakan CSS, sebenarnya kita tidak harus mengatur satu-satu tag <td> sebagaimana contoh di atas (yang memang sama sekali tidak efektif). Berikut adalah modifikasi tampilan tabel menggunakan **internal style** CSS:

index.html

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Belajar Tabel HTML</title>
6      <style>
7          table{
8              border: 3px solid black;
9          }
10         td {
11             border: 2px solid black;
12         }
13     </style>
14 </head>
15 <body>
16 <h3>CSS, property border = 3px (table) dan 2px (td)</h3>
17 <table>
18     <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
19     <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
20 </table>
21 </body>
22 </html>

```

Seperti yang terlihat, penggunaan CSS akan membuat kode HTML menjadi ‘bersih’, karena kita memindahkan seluruh pengaturan tampilan kepada CSS.

10.3 Atribut Rules

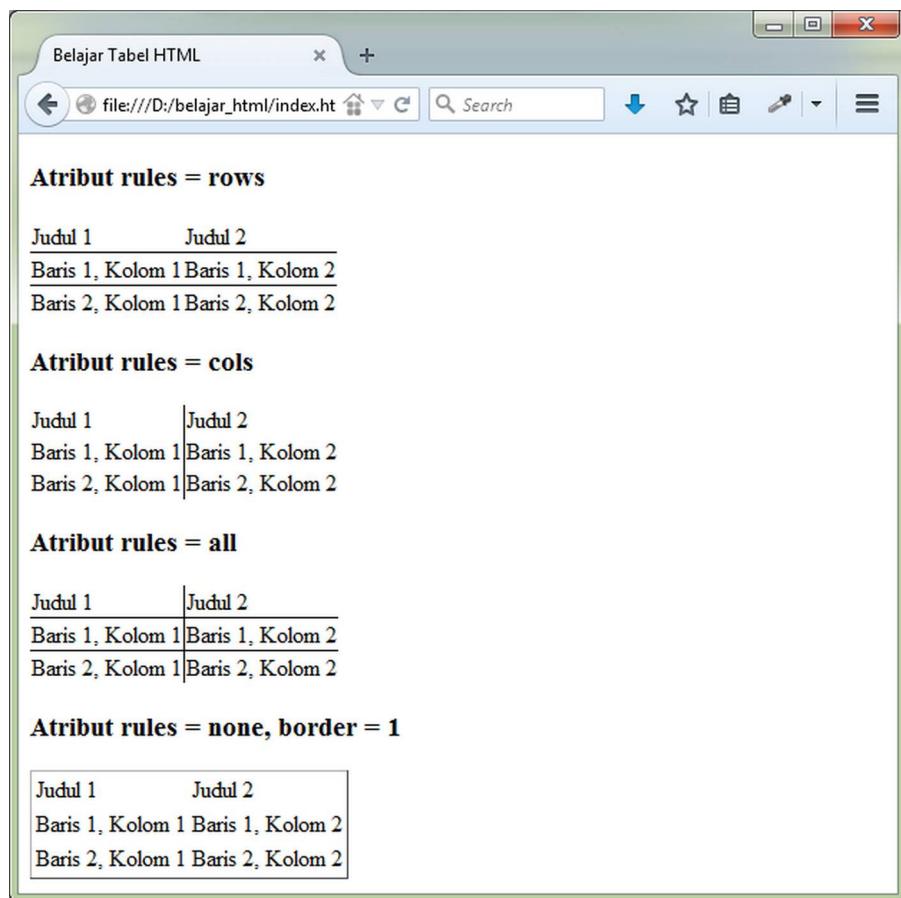
Atribut **border** yang dibahas sebelum ini akan membuat garis pembatas untuk semua sel tabel. Apabila kita ingin lebih spesifik mengatur garis-garis ini (misalkan untuk kolom saja atau baris

saja) bisa menggunakan atribut **rules**. Atribut ini bisa diisi dengan nilai: *cols*, *rows*, *none*, atau *all*.

Berikut contoh penggunaan atribut rules:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Atribut rules = rows</h3>
9 <table rules="rows">
10  <tr><td>Judul 1</td><td>Judul 2</td></tr>
11  <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
12  <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
13 </table>
14
15 <h3>Atribut rules = cols</h3>
16 <table rules="cols">
17  <tr><td>Judul 1</td><td>Judul 2</td></tr>
18  <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
19  <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
20 </table>
21
22 <h3>Atribut rules = all</h3>
23 <table rules="all">
24  <tr><td>Judul 1</td><td>Judul 2</td></tr>
25  <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
26  <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
27 </table>
28
29 <h3>Atribut rules = none, border = 1</h3>
30 <table rules="none" border="1">
31  <tr><td>Judul 1</td><td>Judul 2</td></tr>
32  <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
33  <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
34 </table>
35 </body>
36 </html>
```



Gambar: Penggunaan property rules pada tabel HTML

Akan tetapi, sama seperti atribut **border**, atribut **rules** juga dinyatakan *deprecated* dan tidak disarankan untuk digunakan lagi.

Efek yang sama bisa dihasilkan dengan menggunakan beberapa *property* CSS, seperti contoh berikut ini:

index.html

```

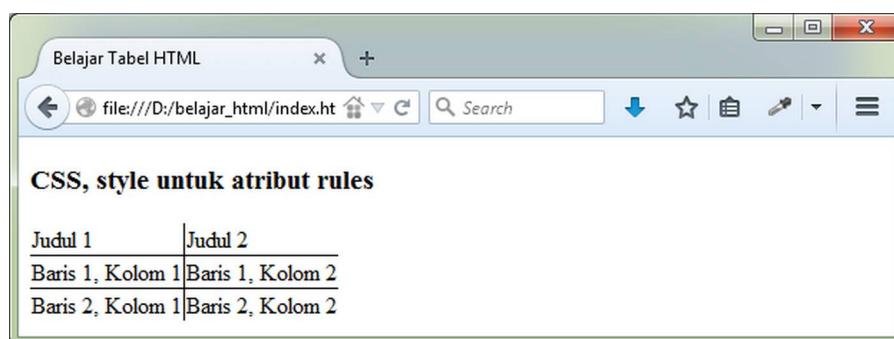
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6   <style>
7     table{
8       border-collapse:collapse;
9       border-spacing:0;
10      border-style:hidden;
11    }
12    td{
13      border:1px solid black;
14    }

```

```

15  </style>
16 </head>
17 <body>
18 <h3>CSS, style untuk atribut rules</h3>
19 <table>
20 <tr><td>Judul 1</td><td>Judul 2</td></tr>
21 <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
22 <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
23 </table>
24 </body>
25 </html>

```



Gambar: Penggunaan CSS sebagai pengganti atribut rules

10.4 Atribut Cellspacing

Untuk mengatur jarak antara 1 sel dengan sel lain di sekelilingnya, bisa dilakukan dengan atribut **cellspacing**. Nilai untuk atribut *cellspacing* berupa angka dengan satuan pixel. Agar lebih jelas, langsung saja kita masuk kepada contoh kode HTML-nya:

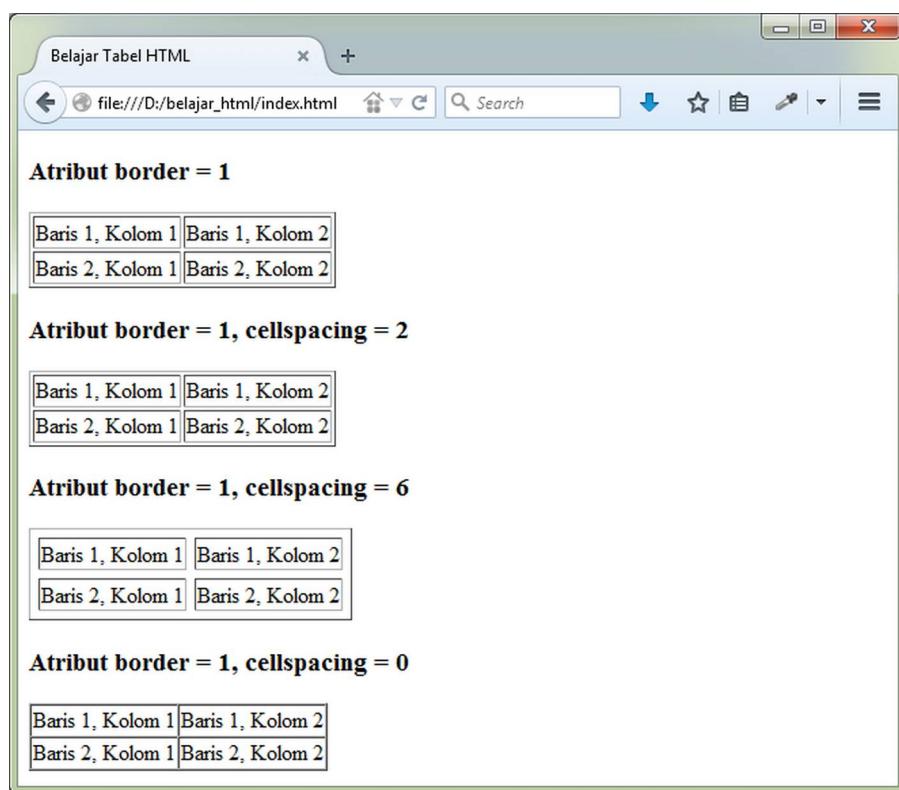
index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Atribut border = 1</h3>
9 <table border="1">
10  <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
11  <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
12 </table>
13
14 <h3>Atribut border = 1, cellspacing = 2 </h3>

```

```
15 <table border="1" cellspacing="2">
16   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
17   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
18 </table>
19
20 <h3>Atribut border = 1, cellspacing = 6 </h3>
21 <table border="1" cellspacing="6">
22   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
23   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
24 </table>
25
26 <h3>Atribut border = 1, cellspacing = 0 </h3>
27 <table border="1" cellspacing="0">
28   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
29   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
30 </table>
31 </body>
32 </html>
```



Gambar: Penggunaan atribut cellspacing pada tabel HTML

Dalam contoh di atas saya menggunakan beberapa atribut **cellspacing** untuk melihat efeknya. Dapat anda lihat bahwa secara default, web browser menggunakan nilai `cellspacing="2"` jika atribut ini tidak ditulis (seperti pada tabel pertama dan kedua).

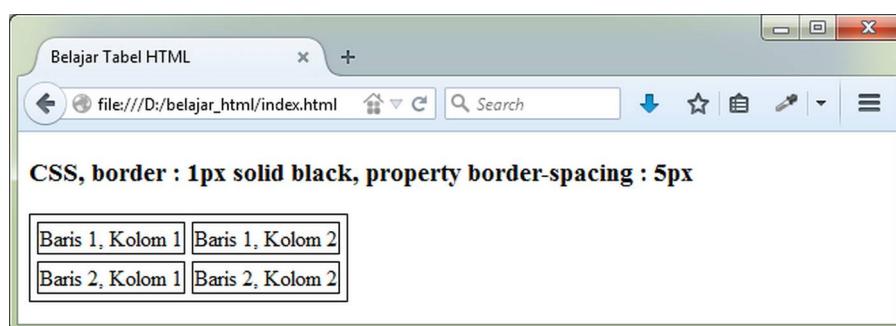
Pada tabel ketiga saya menggunakan `cellspacing="6"`, sehingga akan menghasilkan lebar antar sel tabel sebesar 6 pixel. Pada tabel terakhir, saya bisa mendapatkan efek tabel yang ‘bersih’ dengan denga men-set `cellspacing` menjadi “0”.

Sama seperti atribut `border`, atribut `cellspacing` juga dinyatakan *deprecated*.

Dengan menggunakan CSS, atribut `cellspacing` bisa diganti dengan property `border-spacing`, seperti contoh berikut ini:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6   <style>
7     table {
8       border: 1px solid black;
9       border-spacing: 5px
10    }
11   td {
12     border: 1px solid black
13   }
14 </style>
15 </head>
16 <body>
17 <h3>CSS, border : 1px solid black, property border-spacing : 5px</h3>
18 <table>
19   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
20   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
21 </table>
22 </body>
23 </html>
```



Gambar: Penggunaan CSS sebagai pengganti atribut cellspacing

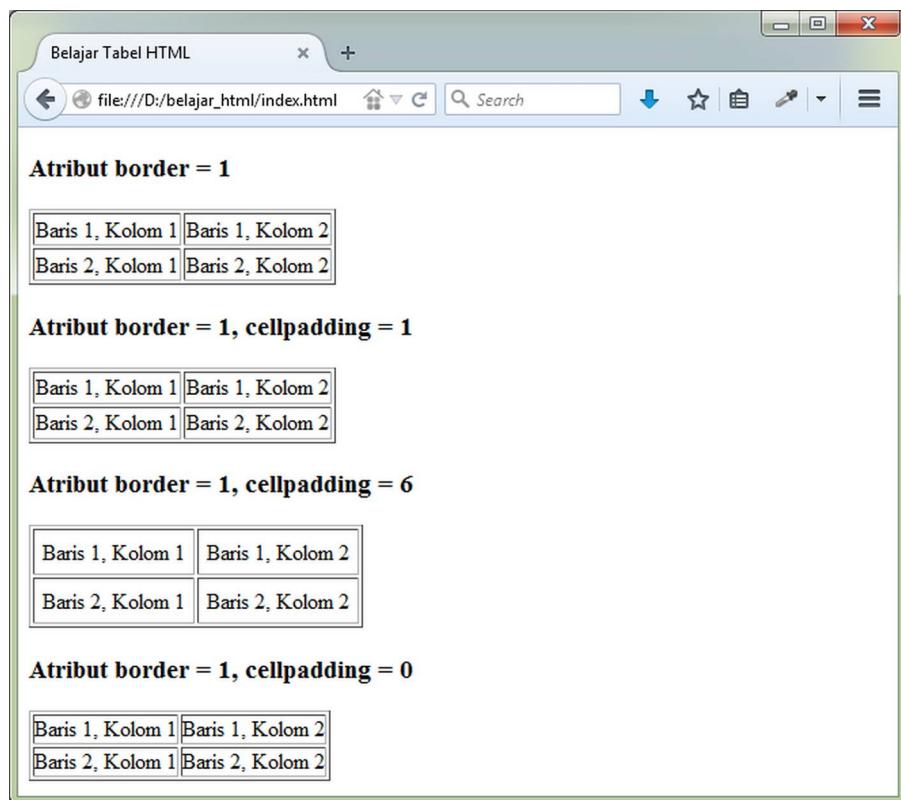
10.5 Atribut Cellpadding

Selain mengatur jarak antar sel, kita juga bisa mengatur jarak antara garis border sel dengan teks. Untuk keperluan ini HTML menyediakan atribut **cellpadding**. Nilai dari atribut **cellpadding** adalah angka dalam satuan pixel.

Berikut contoh penggunaan atribut **cellpadding**:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8
9 <h3>Atribut border = 1</h3>
10 <table border="1">
11   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
12   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
13 </table>
14
15 <h3>Atribut border = 1, cellpadding = 1</h3>
16 <table border="1" cellpadding="1">
17   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
18   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
19 </table>
20
21 <h3>Atribut border = 1, cellpadding = 6</h3>
22 <table border="1" cellpadding="6">
23   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
24   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
25 </table>
26
27 <h3>Atribut border = 1, cellpadding = 0</h3>
28 <table border="1" cellpadding="0">
29   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
30   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
31 </table>
32
33 </body>
34 </html>
```



Gambar: Penggunaan atribut cellpadding pada tabel HTML

Dalam contoh di atas saya membuat beberapa nilai untuk **cellpadding**. Seperti yang terlihat pada tabel pertama dan kedua, jika atribut **cellpadding** tidak ditulis, web browser menggunakan nilai **cellpadding="1"**.

Sama seperti atribut-atribut sebelumnya, **cellpadding** juga telah dinyatakan *deprecated*. Menggunakan CSS, efek yang sama bisa dihasilkan dengan property **padding** yang ditempatkan pada tag **<td>**.

Berikut contoh penggunaannya:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>CSS, border: 1px solid black (table), padding: 5px (td)</h3>
9 <table style="border: 1px solid black">
10  <tr>
11    <td style="border: 1px solid black; padding:5px">
12      Baris 1, Kolom 1</td>
13    <td style="border: 1px solid black; padding:5px">

```

```

14     Baris 1, Kolom 2</td>
15 </tr>
16 <tr>
17     <td style="border: 1px solid black; padding:5px">
18     Baris 2, Kolom 1</td>
19     <td style="border: 1px solid black; padding:5px">
20     Baris 2, Kolom 2</td>
21 </tr>
22 </table>
23 </body>
24 </html>

```



Gambar: Penggunaan CSS sebagai pengganti atribut cellpadding

Dalam kode di atas saya menggunakan **inline style** CSS, sehingga terlihat ‘kurang rapi’. Jika menggunakan **internal style** CSS, struktur tabel menjadi ‘bersih’ karena seluruh pengaturan tampilan tabel dipindahkan ke dalam tag `<style>`. Berikut contohnya:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="UTF-8">
5     <title>Belajar Tabel HTML</title>
6     <style>
7         table {
8             border:2px solid black;
9         }
10        td {
11            padding:10px;
12            border:1px solid black;
13        }
14     </style>
15 </head>
16 <body>
17     <h3>CSS, border: 2px solid black (table), padding: 10px (td)</h3>
18     <table>

```

```
19  <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
20  <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
21 </table>
22 </body>
23 </html>
```

10.6 Atribut Bgcolor

Pengaturan warna background tabel di dalam HTML sebaiknya menggunakan CSS, akan tetapi saya juga akan membahas cara penggunaannya menggunakan atribut **bgcolor**, karena inilah cara paling cepat. Atribut ini sudah dinyatakan *deprecated* dan sangat disarankan untuk beralih ke CSS.

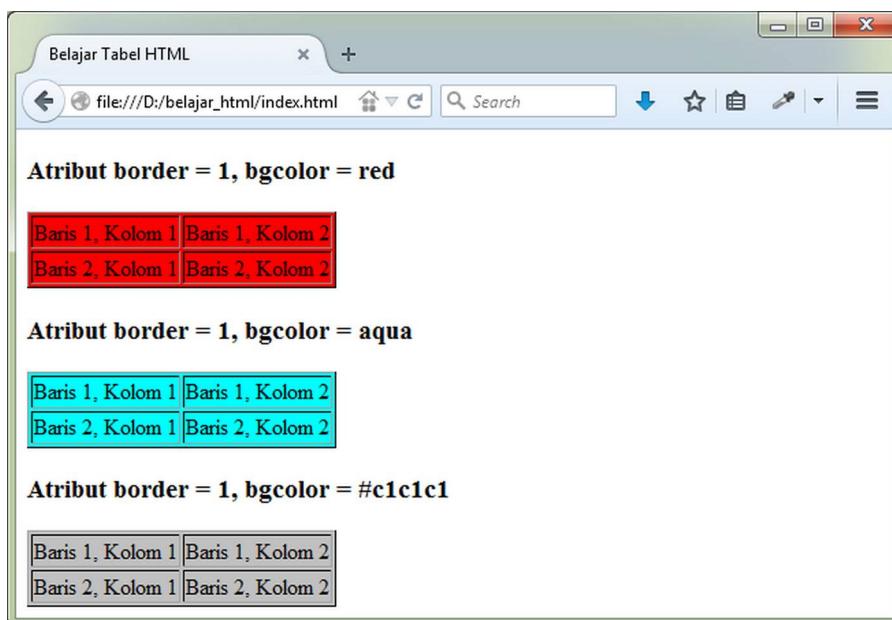
Nilai dari atribut **bgcolor** bisa berupa nama warna (dalam bahasa inggris), atau nilai warna dalam format *hexadesimal* (dengan awalan karakter # dan diikuti 6 digit nilai warna RBG).

Berikut adalah contoh penggunaan atribut **bgcolor**:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8
9 <h3>Atribut border = 1, bgcolor = red</h3>
10 <table border="1" bgcolor="red">
11   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
12   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
13 </table>
14
15 <h3>Atribut border = 1, bgcolor = aqua</h3>
16 <table border="1" bgcolor="aqua">
17   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
18   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
19 </table>
20
21 <h3>Atribut border = 1, bgcolor = #c1c1c1</h3>
22 <table border="1" bgcolor="#c1c1c1">
23   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
24   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
25 </table>
26
```

```
27 </body>
28 </html>
```



Gambar: Penggunaan atribut bgcolor pada tabel HTML

Kita juga bisa mengatur warna untuk setiap sel dengan menempatkan atribut **bgcolor** pada setiap tag `<td>`, seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Atribut border = 1, bgcolor = red, blue, olive dan gray</h3>
9 <table border="1">
10  <tr>
11    <td bgcolor="red">Baris 1, Kolom 1</td>
12    <td bgcolor="blue">Baris 1, Kolom 2</td>
13  </tr>
14  <tr>
15    <td bgcolor="olive">Baris 2, Kolom 1</td>
16    <td bgcolor="gray">Baris 2, Kolom 2</td>
17  </tr>
18 </table>
19 </body>
20 </html>
```

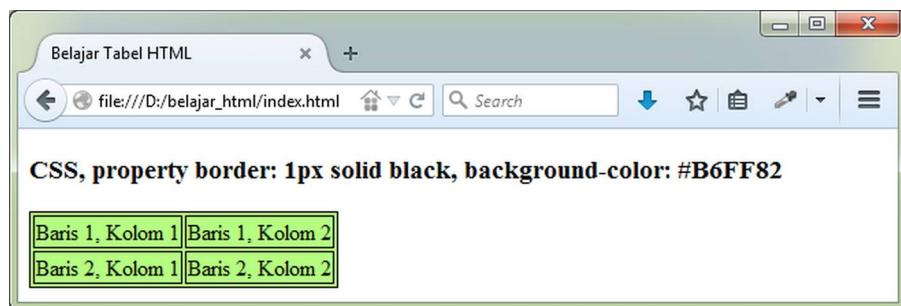


Gambar: Penggunaan atribut bgcolor pada tag <td>

Dengan menggunakan CSS, pengaturan warna background dilakukan dengan property `background-color`, seperti contoh contoh berikut ini:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6   <style>
7     table {
8       border: 1px solid black;
9       background-color:#B6FF82
10    }
11    td {
12      border: 1px solid black
13    }
14  </style>
15 </head>
16 <body>
17 <h3>CSS, property border: 1px solid black, background-color: #B6FF82</h3>
18 <table>
19   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
20   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
21 </table>
22 </body>
23 </html>
```



Gambar: Penggunaan CSS sebagai pengganti atribut bgcolor

10.7 Atribut Width dan Height

Secara default, web browser akan mengatur lebar dan tinggi tabel tergantung dari banyak teks di dalam sel tabel. Sebagai contoh, perhatikan kode berikut ini:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Atribut border = 1</h3>
9 <table border="1" >
10  <tr>
11    <td>Baris 1, Kolom 1</td>
12    <td>Baris 1, Kolom 2</td>
13  </tr>
14  <tr>
15    <td>Lorem ipsum dolor sit amet, consectetur adipisicing elit,
16      sed do eiusmod tempor incididunt et dolore magna aliqua.</td>
17    <td>Baris 2, Kolom 2</td>
18  </tr>
19 </table>
20 </body>
21 </html>
```



Gambar: Penggunaan atribut width pada tag tabel HTML

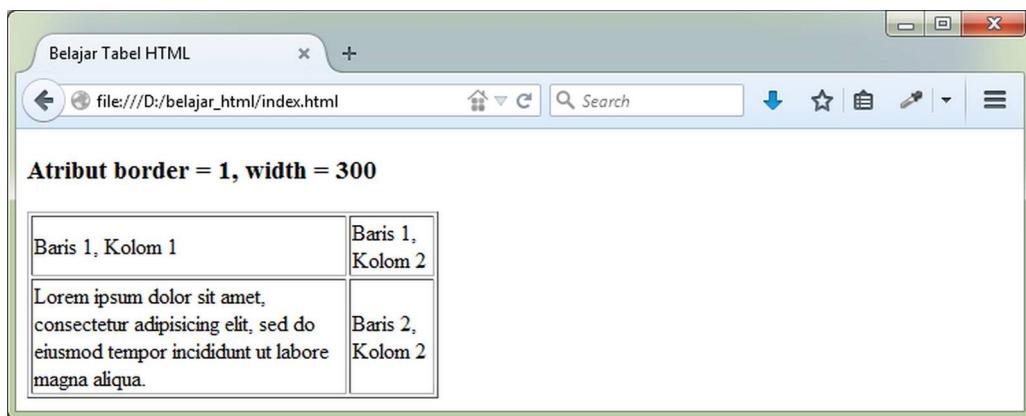
Dalam tabel di atas, isi teks pada baris kedua kolom pertama sangat panjang, sehingga web browser akan membuat lebar kolom 1 sepanjang mungkin agar dapat memuat seluruh teks dalam 1 baris. Jika anda mencoba mengubah ukuran layar web browser, lebar tabel juga akan ikut disesuaikan.

Apabila kita ingin lebar dan tinggi tabel ‘tetap’ dalam ukuran tertentu, bisa menggunakan atribut **width** dan **height**.

Atribut **width** digunakan untuk mengatur lebar tabel dalam satuan *pixel* maupun dalam satuan persen (%). Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Tabel HTML</title>
6   </head>
7   <body>
8     <h3>Atribut border = 1, width = 300</h3>
9     <table border="1" width="300">
10       <tr>
11         <td>Baris 1, Kolom 1</td>
12         <td>Baris 1, Kolom 2</td>
13       </tr>
14       <tr>
15         <td>Lorem ipsum dolor sit amet, consectetur adipisicing elit,
16           sed do eiusmod tempor incididunt ut labore magna aliqua.</td>
17         <td>Baris 2, Kolom 2</td>
18       </tr>
19     </table>
20   </body>
21 </html>
```



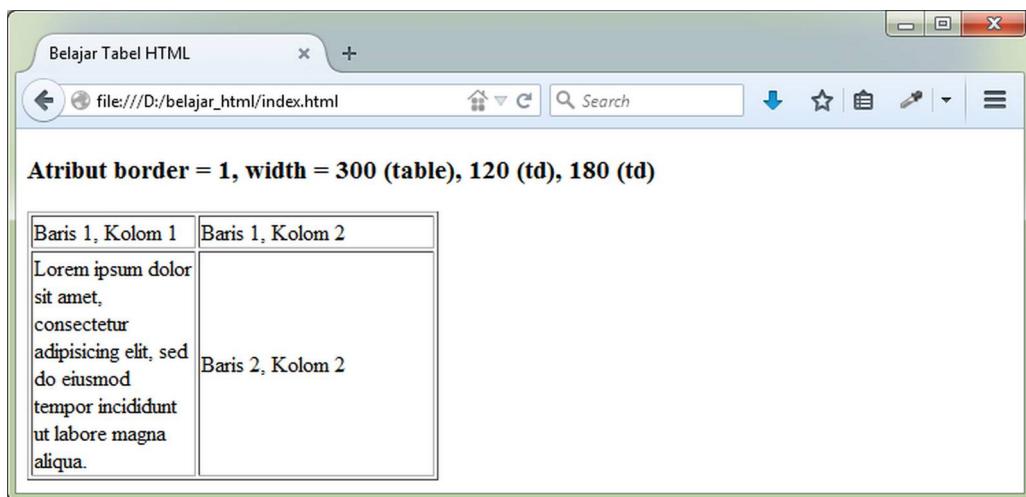
Gambar: Penggunaan atribut width dengan satuan persen

Dalam contoh di atas, saya menempatkan atribut **width** pada tag `<table>` sehingga lebar tabel akan diset sebesar 300 pixel. Namun lebar dari masing-masing kolom diatur secara otomatis oleh web browser.

Jika anda juga ingin mengatur lebar dari masing-masing kolom, tambahkan atribut **width** pada tag `<td>` pertama, seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Atribut border = 1, width = 300 (table), 120 (td), 180 (td)</h3>
9 <table border="1" width="300">
10   <tr>
11     <td width="120">Baris 1, Kolom 1</td>
12     <td width="180">Baris 1, Kolom 2</td>
13   </tr>
14   <tr>
15     <td>Lorem ipsum dolor sit amet, consectetur adipisicing elit,
16       sed do eiusmod tempor incididunt ut labore magna aliqua.</td>
17     <td>Baris 2, Kolom 2</td>
18   </tr>
19 </table>
20 </body>
21 </html>
```



Gambar: Penggunaan atribut width pada tag <td>

Selain menggunakan *pixel*, kita juga bisa menggunakan satuan persen.

Apabila kita membuat atribut width = "50%" maka tabel akan di-set sebesar 50% dari tag induknya (mengenai tag induk/*parent tag* telah kita bahas pada bab 2, bagian **DOM tree**).

Berikut contoh penggunaan satuan persen untuk mengatur lebar tabel:

index.html

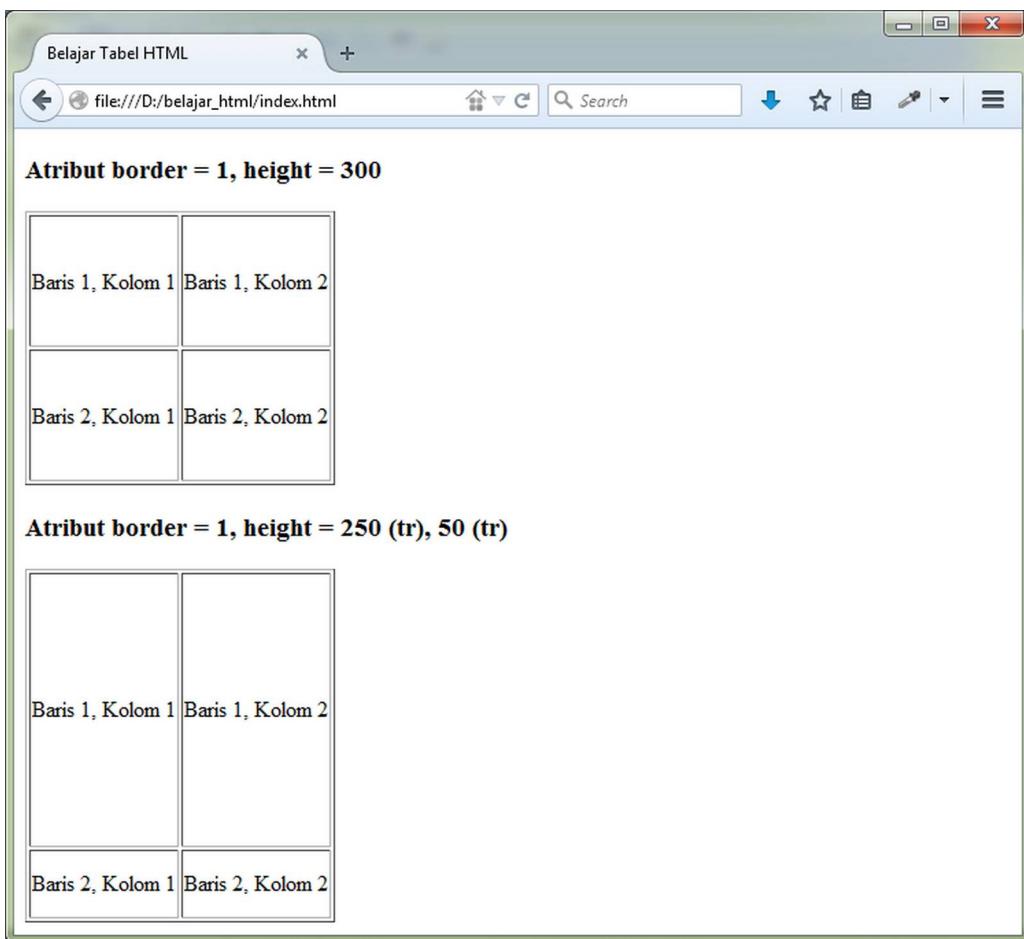
```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Atribut border = 1, width = 50% (table), 25% (td), 75% (td)</h3>
9 <table border="1" width="50%">
10  <tr>
11    <td width="25%">Baris 1, Kolom 1</td>
12    <td width="75%">Baris 1, Kolom 2</td>
13  </tr>
14  <tr>
15    <td>Lorem ipsum dolor sit amet, consectetur adipisicing elit,
16      sed do eiusmod tempor incididunt ut labore magna aliqua.</td>
17    <td>Baris 2, Kolom 2</td>
18  </tr>
19 </table>
20 </body>
21 </html>
```

Dalam contoh di atas, lebar tabel akan ditampilkan sebesar 50% dari lebar web browser (parent tag <table> adalah tag <body>, dimana secara default berukuran sebesar jendela web browser).

Untuk mengatur tinggi dari tabel, bisa menggunakan atribut **height**. Seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8
9 <h3>Atribut border = 1, height = 300</h3>
10 <table border="1" height="200">
11   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
12   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
13 </table>
14
15 <h3>Atribut border = 1, height = 250 (tr), 50 (tr)</h3>
16 <table border="1" >
17   <tr height="200">
18     <td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td>
19   </tr>
20   <tr height="50">
21     <td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td>
22   </tr>
23 </table>
24
25 </body>
26 </html>
```



Gambar: Penggunaan atribut height pada tag tabel HTML

Cara penggunaan atribut **height** hampir sama dengan atribut **width**, namun untuk mengatur tinggi baris, kita meletakkannya di dalam tag <tr>.

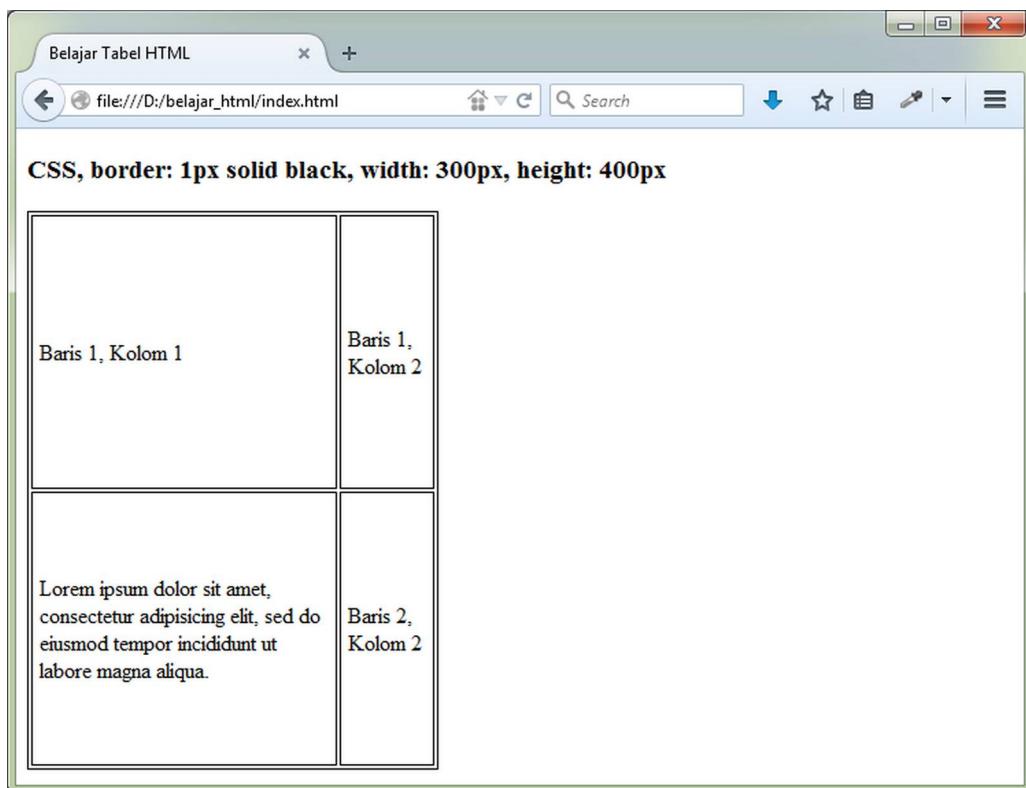


Penggunaan atribut **height** untuk tabel tidak terlalu sering digunakan. Salah satu tujuan atribut width adalah untuk membuat layout halaman seperti *header*, *footer*, *sidebar*, dll.

Dalam HTML5, atribut **width** dan **height** sudah dinyatakan *deprecated*. Hal yang sama sebaiknya dilakukan menggunakan CSS melalui property **width** dan **height**. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6   <style>
7     table {
8       border:1px solid black;
9       width: 300px;
10      height: 400px;
11    }
12    tr {
13      height: 200px;
14    }
15    td {
16      padding:5px;
17      border:1px solid black;
18    }
19  </style>
20 </head>
21 <body>
22 <h3>CSS, border: 1px solid black, width: 300px, height: 400px</h3>
23 <table>
24   <tr >
25     <td>Baris 1, Kolom 1</td>
26     <td>Baris 1, Kolom 2</td>
27   </tr>
28   <tr>
29     <td>Lorem ipsum dolor sit amet, consectetur adipisicing elit,
30           sed do eiusmod tempor incididunt ut labore magna aliqua.</td>
31     <td>Baris 2, Kolom 2</td>
32   </tr>
33 </table>
34 </body>
35 </html>
```



Gambar: Penggunaan CSS sebagai pengganti atribut width dan height

Dalam contoh di atas, saya mengatur lebar tabel sebesar 300 pixel dan tinggi sebesar 400 pixel. Tinggi setiap baris juga di-set sebesar 200 px.

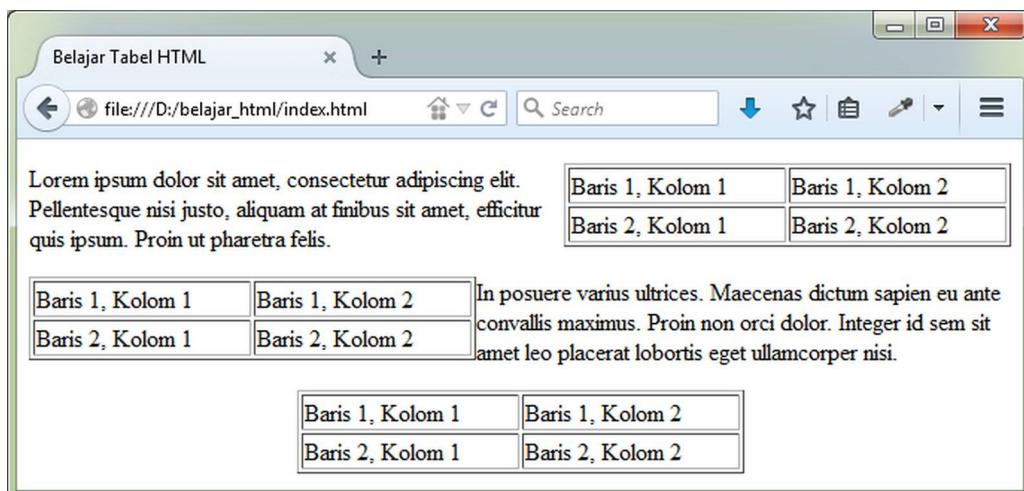
10.8 Atribut Align

Secara default, tabel akan ditampilkan pada baris baru di sisi kiri jendela web browser. Untuk mengubah posisi tabel, bisa menggunakan atribut **align**. Nilai dari atribut ini adalah: *left*, *center* dan *right*. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8
9 <table border="1" width="300" align="right">
10   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
11   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
12 </table>
```

```
13
14 <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.
15 Pellentesque nisi justo, aliquam at finibus sit amet,
16 efficitur quis ipsum. Proin ut pharetra felis.</p>
17
18 <table border="1" width="300" align="left">
19   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
20   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
21 </table>
22
23 <p>In posuere varius ultrices. Maecenas dictum sapien eu
24 ante convallis maximus. Proin non orci dolor. Integer
25 id sem sit amet leo placerat lobortis eget ullamcorper nisi.</p>
26
27 <table border="1" width="300" align="center">
28   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
29   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
30 </table>
31
32 </body>
33 </html>
```



Gambar: Penggunaan atribut align pada tabel HTML

Atribut **align** juga sudah dinyatakan *deprecated* dan kita disarankan beralih menggunakan property **float** dan **margin** melalui CSS. Berikut contohnya:

index.html

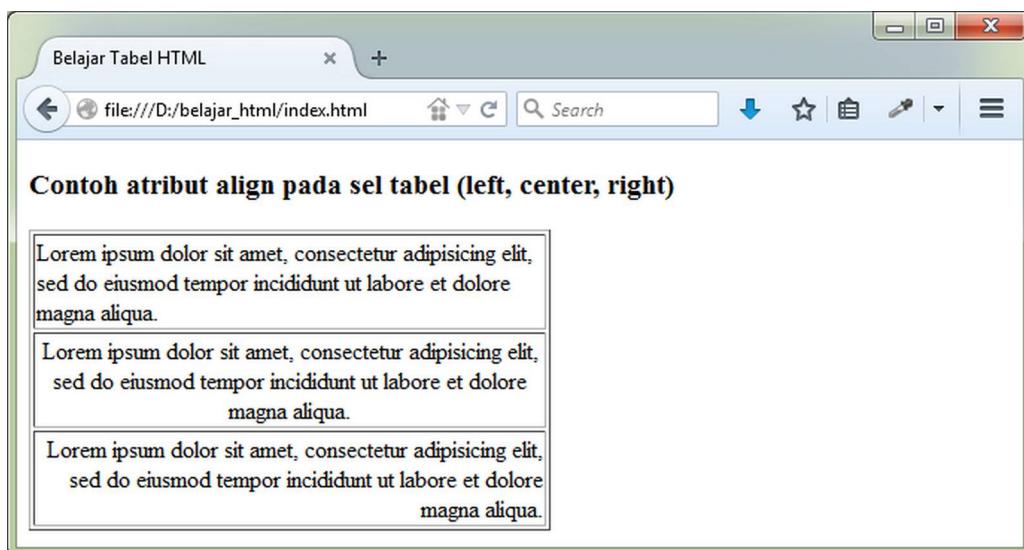
```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8
9 <table border="1" width="300" style="float:right">
10  <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
11  <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
12 </table>
13
14 <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.
15 Pellentesque nisi justo, aliquam at finibus sit amet,
16 efficitur quis ipsum. Proin ut pharetra felis.</p>
17
18 <table border="1" width="300" style="float:left">
19  <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
20  <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
21 </table>
22
23 <p>In posuere varius ultrices. Maecenas dictum sapien eu
24 ante convallis maximus. Proin non orci dolor. Integer
25 id sem sit amet leo placerat lobortis eget ullamcorper nisi.</p>
26
27 <table border="1" width="300" style="margin:auto">
28  <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
29  <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
30 </table>
31
32 </body>
33 </html>
```

Hasil kode di atas sama persis dengan hasil yang didapat jika menggunakan atribut align.

Jika atribut align ditempatkan pada tag `<td>`, maka fungsinya beralih untuk mengatur rata teks yang berada di dalam sel tabel tersebut, berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Contoh atribut align pada sel tabel (left, center, right)</h3>
9 <table border="1" width="350">
10  <tr>
11    <td align="left">Lorem ipsum dolor sit amet, consectetur
12      adipisicing elit, sed do eiusmod tempor incididunt
13      ut labore et dolore magna aliqua.</td>
14  </tr>
15  <tr>
16    <td align="center">Lorem ipsum dolor sit amet, consectetur
17      adipisicing elit, sed do eiusmod tempor incididunt
18      ut labore et dolore magna aliqua.</td>
19  </tr>
20  <tr>
21    <td align="right">Lorem ipsum dolor sit amet, consectetur
22      adipisicing elit, sed do eiusmod tempor incididunt
23      ut labore et dolore magna aliqua.</td>
24  </tr>
25 </table>
26 </body>
27 </html>
```



Gambar: Penggunaan atribut align pada tag <td> tabel HTML

Penggunaan atribut align pada tag <td> ini juga sudah *deprecated*. Kita bisa menggantinya dengan property **text-align** menggunakan CSS, seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Tabel HTML</title>
6   </head>
7   <body>
8     <h3>Contoh text-align CSS pada sel tabel (left, center, right)</h3>
9     <table border="1" width="350">
10       <tr>
11         <td style="text-align:left">Lorem ipsum dolor sit amet, consectetur
12           adipisicing elit, sed do eiusmod tempor incididunt
13           ut labore et dolore magna aliqua.</td>
14       </tr>
15       <tr>
16         <td style="text-align:center">Lorem ipsum dolor sit amet, consectetur
17           adipisicing elit, sed do eiusmod tempor incididunt
18           ut labore et dolore magna aliqua.</td>
19       </tr>
20       <tr>
21         <td style="text-align:right">Lorem ipsum dolor sit amet, consectetur
22           adipisicing elit, sed do eiusmod tempor incididunt
23           ut labore et dolore magna aliqua.</td>
24       </tr>
25     </table>
26   </body>
27 </html>
```

10.9 Atribut Valign

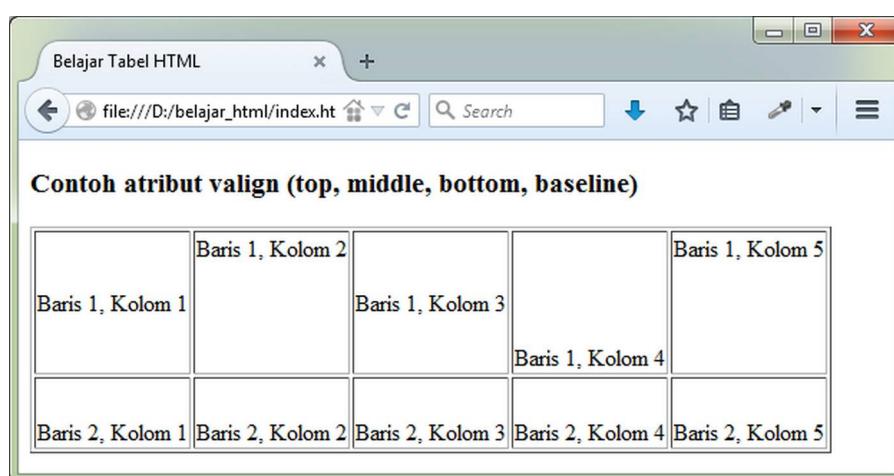
Apabila kita mengatur tinggi kolom dengan atribut **height**, secara default web browser memposisikan teks berada di tengah sel tabel (secara vertikal). Untuk mengatur posisi ini, bisa menggunakan atribut **valign**. Nilai atribut ini adalah salah satu dari nilai: *baseline*, *middle*, *top*, dan *bottom*.

Kita bisa menempatkan atribut **valign** pada tag <tr> untuk mengatur seluruh baris, atau meletakkannya pada tag <td> untuk mengatur setiap sel secara terpisah.

Berikut contoh penggunaan atribut valign:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Contoh atribut valign (top, middle, bottom, baseline)</h3>
9 <table border="1" >
10  <tr height="100">
11    <td>Baris 1, Kolom 1</td>
12    <td valign="top">Baris 1, Kolom 2</td>
13    <td valign="middle">Baris 1, Kolom 3</td>
14    <td valign="bottom">Baris 1, Kolom 4</td>
15    <td valign="baseline">Baris 1, Kolom 5</td>
16  </tr>
17  <tr height="50" valign="bottom">
18    <td>Baris 2, Kolom 1</td>
19    <td>Baris 2, Kolom 2</td>
20    <td>Baris 2, Kolom 3</td>
21    <td>Baris 2, Kolom 4</td>
22    <td>Baris 2, Kolom 5</td>
23  </tr>
24 </table>
25 </body>
26 </html>
```



Gambar: Penggunaan atribut valign pada tabel HTML

Dalam HTML5, atribut **valign** sudah dinyatakan *deprecated*, dan kita sebaiknya menggunakan CSS melalui property **vertical-align**. Berikut contohnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Contoh vertical-align CSS (top, middle, bottom, baseline)</h3>
9 <table border="1" >
10  <tr style="height:100px">
11    <td>Baris 1, Kolom 1</td>
12    <td style="vertical-align:top">Baris 1, Kolom 2</td>
13    <td style="vertical-align:middle">Baris 1, Kolom 3</td>
14    <td style="vertical-align:bottom">Baris 1, Kolom 4</td>
15    <td style="vertical-align:baseline">Baris 1, Kolom 5</td>
16  </tr>
17  <tr style="height:50px; vertical-align:bottom">
18    <td>Baris 2, Kolom 1</td>
19    <td>Baris 2, Kolom 2</td>
20    <td>Baris 2, Kolom 3</td>
21    <td>Baris 2, Kolom 4</td>
22    <td>Baris 2, Kolom 5</td>
23  </tr>
24 </table>
25 </body>
26 </html>
```



Nilai **baseline** akan bergantung kepada *parent tag*. Karena disini saya tidak menggunakan parent tag untuk tabel, maka tampilannya mirip dengan nilai **top**.

10.10 Atribut Rowspan dan Colspan

Dalam menampilkan data yang rumit, kadang kita perlu menggabungkan beberapa sel tabel menjadi sel yang lebih besar. Untuk keperluan ini, HTML menyediakan atribut **rowspan** dan **colspan**. Atribut **rowspan** digunakan untuk menggabungkan sel tabel dalam bentuk baris, sedangkan atribut **colspan** digunakan untuk membentuk kolom yang lebih panjang. Kedua atribut ini ditempatkan pada tag **<td>**.

Agar lebih mudah memahaminya, langsung saja kita masuk dengan contoh kode program:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Contoh atribut colspan dan rowspan</h3>
9 <table border="1">
10   <tr>
11     <td>Baris 1, Kolom 1</td>
12     <td>Baris 1, Kolom 2</td>
13     <td>Baris 1, Kolom 3</td>
14   </tr>
15   <tr>
16     <td>Baris 2, Kolom 1</td>
17     <td colspan="2" >Baris 2, Kolom 2 & 3</td>
18   </tr>
19   <tr>
20     <td rowspan="2" > Baris 3 & 4, Kolom 1</td>
21     <td> Baris 3, Kolom 2</td>
22     <td> Baris 3, Kolom 3</td>
23   </tr>
24   <tr>
25     <td> Baris 4, Kolom 2</td>
26     <td> Baris 4, Kolom 3</td>
27   </tr>
28 </table>
29 </body>
30 </html>
```



Gambar: Penggunaan atribut colspan dan rowspan pada tabel HTML

Pada baris ke-17 saya menambahkan atribut `colspan="2"` ke dalam tag `<td>`. Atribut ini akan membuat tag `<td>` ‘membesar’ dan mengambil tempat dimana seharusnya memerlukan 2 sel tabel. Jika anda ingin tag `<td>` menempati 3 sel, maka tinggal menggunakan atribut `colspan="3"`.

Pada baris 20 saya membuat sel tabel bergabung dengan baris dibawahnya dengan atribut `rowspan="2"`, sama seperti atribut `colspan`, kita bisa mengubah nilai dari atribut ini jika butuh ruang baris yang lebih lebar.

Sebagai contoh lain, dalam kode berikut saya membuat tabel yang umum digunakan dalam menampilkan data dengan menggabungkan beberapa sel tabel pada baris pertama. Berikut contohnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Contoh atribut colspan dan rowspan</h3>
9 <table border="1">
10  <tr>
11    <td rowspan="2">No.</td>
12    <td colspan="3" style="text-align:center">Judul Kolom</td>
13  </tr>
14  <tr>
15    <td>Sub Judul 1</td><td>Sub Judul 2</td><td>Sub Judul 3</td>
16  </tr>
17  <tr>
18    <td>1. </td><td>Data 1</td><td>Data 2</td><td>Data 3</td>
19  </tr>
20  <tr>
21    <td>2. </td><td>Data 1</td><td>Data 2</td><td>Data 3</td>
22  </tr>
23 </table>
24 </body>
25 </html>
```



Gambar: Penggunaan atribut colspan dan rowspan pada tabel HTML

Dalam tabel di atas saya menyatukan 3 kolom pada baris pertama untuk menjadi judul kolom, dan menyatukan baris pertama dan kedua untuk penomoran. Struktur tabel seperti ini umum digunakan dalam penyajian data.

10.11 Th Element

Walaupun kita bisa membuat judul kolom menggunakan tag <td> seperti pada contoh sebelumnya, HTML menyediakan tag khusus untuk membuat struktur judul kolom, yakni tag <th> (*table header*). Tag ini digunakan sebagai pengganti tag <td> yang berada pada baris pertama tabel.

Berikut contoh penggunaan tag <th>:

index.html

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Belajar Tabel HTML</title>
6  </head>
7  <body>
8  <h3>Contoh penggunaan tag th</h3>
9  <table border="1">
10     <tr><th>Judul 1</th><th>Judul 2</th></tr>
11     <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
12     <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
13 </table>
14 </body>
15 </html>
```



Gambar: Penggunaan tag <th> pada tabel HTML

Seperti yang terlihat, secara default web browser akan membedakan tampilan tag <th>, yakni membuat teks menjadi tebal (*bold*) dan berada di tengah sel tabel (*center*).

Seluruh atribut (dan juga style CSS) yang berlaku untuk tag <td> juga bisa ditempatkan pada tag <th>.

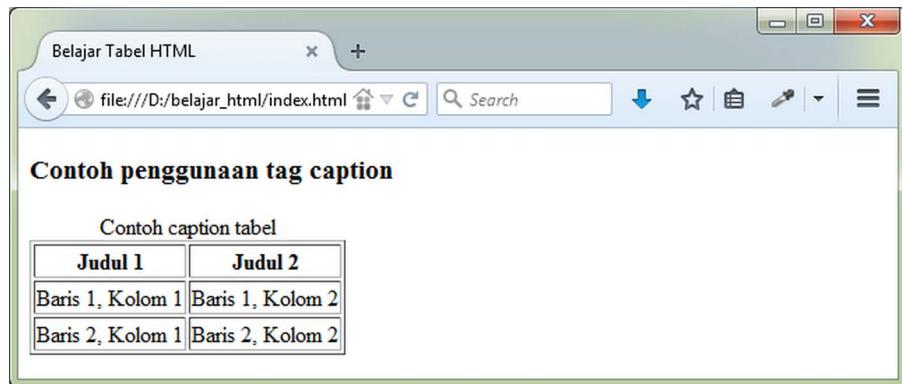
10.12 Caption Element

HTML menyediakan tag <caption> yang bisa ditambahkan ke dalam struktur tabel untuk membuat judul / keterangan tabel. Tag ini ditulis setelah tag pembuka <table>, namun sebelum tag <tr> pertama.

Berikut contoh penggunaan tag <caption>:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Contoh penggunaan tag caption</h3>
9 <table border="1">
10 <caption>Caption Tabel Disini</caption>
11   <tr><th>Judul 1</th><th>Judul 2</th></tr>
12   <tr><td>Baris 1, Kolom 1</td><td>Baris 1, Kolom 2</td></tr>
13   <tr><td>Baris 2, Kolom 1</td><td>Baris 2, Kolom 2</td></tr>
14 </table>
15 </body>
16 </html>
```



Gambar: Penggunaan tag `<caption>` pada tabel HTML

Walaupun pada contoh di atas saya membuat tag `<caption>` di dalam tabel, tetapi web browser menampilkannya di atas tabel. Jika anda ingin judul/keterangan ini berada di bawah tabel, bisa menambahkan atribut `align`, seperti contoh berikut:

```
<caption align="bottom">Caption Tabel Disini</caption>
```

Atribut `align` bisa bernilai: *top* (default), *bottom*, *right* atau *left*. Namun dalam HTML5 atribut ini sudah *deprecated*, sehingga kita bisa menggantinya dengan property `caption-side` CSS, seperti contoh berikut:

```
<caption style="caption-side:bottom">Caption Tabel Disini</caption>
```

10.13 Colgroup dan Col Element

Jika anda perhatikan, selain sebagai struktur, tag `<tr>` juga berfungsi sebagai ‘container’ dari suatu baris. Jika kita ingin mengubah warna background untuk seluruh sel tabel pada baris yang sama, kita tinggal menempatkan *atribut/style* CSS pada tag `<tr>`. Namun tidak demikian untuk kolom.

Dalam struktur tabel yang telah kita pelajari hingga saat ini, tidak terdapat ‘container’ untuk kolom. Sehingga, apabila ingin mengubah warna background untuk sebuah kolom tabel, terpaksa harus di-set secara manual pada setiap sel.

Untuk mengatasi hal ini, HTML menyediakan tag `<colgroup>` dan tag `<col>`. Kedua tag ini berfungsi sebagai ‘pengait’ untuk seluruh kolom. Tag ini ditulis sebelum tag `<tr>` pertama.

Berikut contoh penggunaan tag `<colgroup>` dan tag `<col>`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Contoh penggunaan tag colgroup dan tag col tabel</h3>
9 <table border="1">
10  <colgroup>
11    <col>
12    <col>
13  </colgroup>
14  <tr>
15    <th>Judul 1</th>
16    <th>Judul 2</th>
17  </tr>
18  <tr>
19    <td>Baris 1, Kolom 1</td>
20    <td>Baris 1, Kolom 2</td>
21  </tr>
22  <tr>
23    <td>Baris 2, Kolom 1</td>
24    <td>Baris 2, Kolom 2</td>
25  </tr>
26 </table>
27 </body>
28 </html>
```



Gambar: Penggunaan tag `<colgroup>` dan `<col>` pada tabel HTML

Seperti yang terlihat, tag `<colgroup>` dan tag `<col>` tidak berdampak apa-apa untuk tampilan. Kedua tag ini baru berfungsi jika menggunakan *atribut* atau *style* CSS.

Sebagai contoh, jika saya ingin memberi warna berlainan pada setiap kolom, saya tinggal

menambahkan atribut *style* kepada tag <col>. Tanpa menggunakan tag ini, kita terpaksa men-set setiap tag <td> untuk kolom yang bersesuaian. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Tabel HTML</title>
6   </head>
7   <body>
8     <h3>Contoh penggunaan tag colgroup dan tag col tabel</h3>
9     <table border="1">
10       <colgroup>
11         <col style="background-color:#C9C9C9">
12         <col style="background-color:#9AF998">
13     </colgroup>
14     <tr>
15       <th>Judul 1</th>
16       <th>Judul 2</th>
17     </tr>
18     <tr>
19       <td>Baris 1, Kolom 1</td>
20       <td>Baris 1, Kolom 2</td>
21     </tr>
22     <tr>
23       <td>Baris 2, Kolom 1</td>
24       <td>Baris 2, Kolom 2</td>
25     </tr>
26   </table>
27 </body>
28 </html>
```



Gambar: Penggunaan tag <colgroup> dan <col> dengan CSS

10.14 Thead, Tbody dan Tfoot Element

HTML masih menyediakan 3 tag lagi untuk membuat struktur lengkap tabel, yakni tag `<thead>`, `<tbody>`, dan `<tfoot>`. Ketiga tag ini merupakan singkatan dari *table head*, *table body*, dan *table foot*.

Sesuai dengan namanya, tag-tag ini digunakan untuk menandakan bagian mana yang berfungsi sebagai judul tabel (*head*), isi tabel (*body*), dan penutup tabel (*foot*). Ketiganya hanya sebagai ‘pembagi’ struktur tabel dan tidak mempengaruhi tampilan tabel secara langsung.

Pembagian tabel seperti ini sangat berguna jika data yang ditampilkan cukup ‘rumit’, atau butuh tempat sebagai lokasi *style* CSS.

Berikut contoh penggunaan tag `<thead>`, `<tbody>`, dan `<tfoot>`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Contoh penggunaan tag thead, tbody dan tfoot</h3>
9 <table border="1">
10  <thead>
11    <tr>
12      <th>No</th>
13      <th>Nama Peserta</th>
14      <th>Alamat</th>
15      <th>Simpanan</th>
16    </tr>
17  </thead>
18  <tfoot>
19    <tr>
20      <td></td>
21      <td>Total</td>
22      <td>---</td>
23      <td>350.000</td>
24    </tr>
25  </tfoot>
26  <tbody>
27    <tr>
28      <td>1.</td>
29      <td>Andi Suryono</td>
30      <td>Jl. Kemerdekaan No.17</td>
31      <td>50.000</td>
```

```

32      </tr>
33      <tr>
34          <td>2.</td>
35          <td>Susi Handayani</td>
36          <td>Jl. Kebangsaan No.225</td>
37              <td>300.000</td>
38      </tr>
39  </tbody>
40 </table>
41 </body>
42 </html>

```



Gambar: Penggunaan tag **<thead>**, **<tbody>** dan **<tfoot>** tabel HTML

Tag **<thead>**, **<tbody>** dan **<tfoot>** di tempatkan sebagai container dari tag **<tr>**. Jika anda perhatikan, saya menempatkan tag **<tfoot>** sebelum tag **<tbody>**, namun web browser menampilkannya pada bagian akhir tabel, ini memang sesuai dengan fungsi tag **<tfoot>** untuk menandakan baris akhir tabel.

Dari tampilan yang dihasilkan, ketiga tag ini tidak menambah efek apapun di dalam tabel. Jadi, apa fungsi tag-tag tabel seperti ini?

Tren perkembangan HTML5 dan berbagai teknologi web akhir-akhir ini sangat menyarankan penggunaan **semantic tag**. Kita didorong untuk menggunakan tag-tag yang ‘bermakna’. Tag **<thead>**, **<tbody>** dan **<tfoot>** adalah bagian dari hal ini.

Walaupun tidak berdampak apa-apa untuk tampilan tabel, ‘mesin’ seperti *search engine* dan web browser masa depan bisa memanfaatkan tag-tag ini untuk menampilkan data dengan lebih baik lagi. Karena itu tidak ada salahnya kita menggunakan struktur tag ini dalam membuat tabel.

Jika anda menggunakan CSS, tag **<thead>**, **<tbody>** dan **<tfoot>** bisa berfungsi sebagai lokasi **selector CSS**, misalkan untuk mengubah style hanya untuk baris terakhir tabel, atau hanya untuk body tabel saja.

10.15 Mengisi Sel Tabel

Dalam contoh-contoh pada bab ini, saya hanya menggunakan teks sederhana sebagai pengisi sel tabel, namun kita bisa memasukkan hampir seluruh tag-tag HTML lain ke dalam sel tabel ini,

termasuk link, gambar, bahkan tabel lain. Berikut contohnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <h3>Contoh isi sel tabel dengan berbagai tag HTML</h3>
9 <table border=1>
10  <tr>
11    <th>No.</th>
12    <th>Alamat Situs</th>
13    <th>Keterangan</th>
14    <th>Screen Shoot</th>
15  </tr>
16  <tr>
17    <td>1</td>
18    <td><a href="www.duniailkom.com">www.duniailkom.com</a></td>
19    <td><p>Situs <em>edukasi</em> programming <b>Indonesia</b></p></td>
20    <td></td>
21  </tr>
22  <tr>
23    <td>2</td>
24    <td><a href="www.google.com">www.google.com</a></td>
25    <td><p>Situs Search Engine <del>No.2</del> No.1 di dunia</p></td>
26    <td></td>
27  </tr>
28 </table>
29 </body>
30 </html>
```



The screenshot shows a web browser window titled "Belajar Tabel HTML". The address bar displays "file:///D:/belajar_html/index.html". The main content area contains a table with the following data:

No.	Alamat Situs	Keterangan	Screen Shoot
1	www.duniaikom.com	Situs edukasi programming Indonesia	
2	www.google.com	Situs Search Engine №.2 No.1 di dunia	

Gambar: Isi sel tabel dengan berbagai tag HTML lainnya

Seperti yang terlihat, pada tabel di atas saya menggunakan berbagai tag yang telah kita pelajari seperti link, format teks, dan gambar pada sel tabel.

10.16 Membuat Layout Menggunakan Tabel

Jika anda telah memahami CSS atau telah mempelajari standar pembuatan web saat ini, judul di atas akan sangat sangat dihindari. Dan saya juga menyarankan anda untuk tidak menggunakan tabel untuk membuat *layout*. *Layout* (tampilan halaman) sebaiknya menggunakan CSS dengan bantuan tag `<div>` atau tag-tag baru HTML5 (akan kita pelajari nantinya).

Walaupun demikian, saya ingin memperlihatkan bagaimana dulu tabel digunakan sedemikian rupa untuk membuat layout halaman. Hal ini lumrah dilakukan sekitar tahun 2000an dimana CSS masih belum terlalu berkembang. Bahkan saya masih ingat ketika pertama kali menggunakan *Macromedia Dreamweaver 8* sekitar tahun 2006 (sebelum diakuisisi oleh *Adobe*). Saat itu aplikasi ini juga akan men-generate layout halaman web menggunakan tabel.

Berikut adalah contoh cara ‘berkreasi’ dengan tabel untuk membuat layout halaman seberhana:

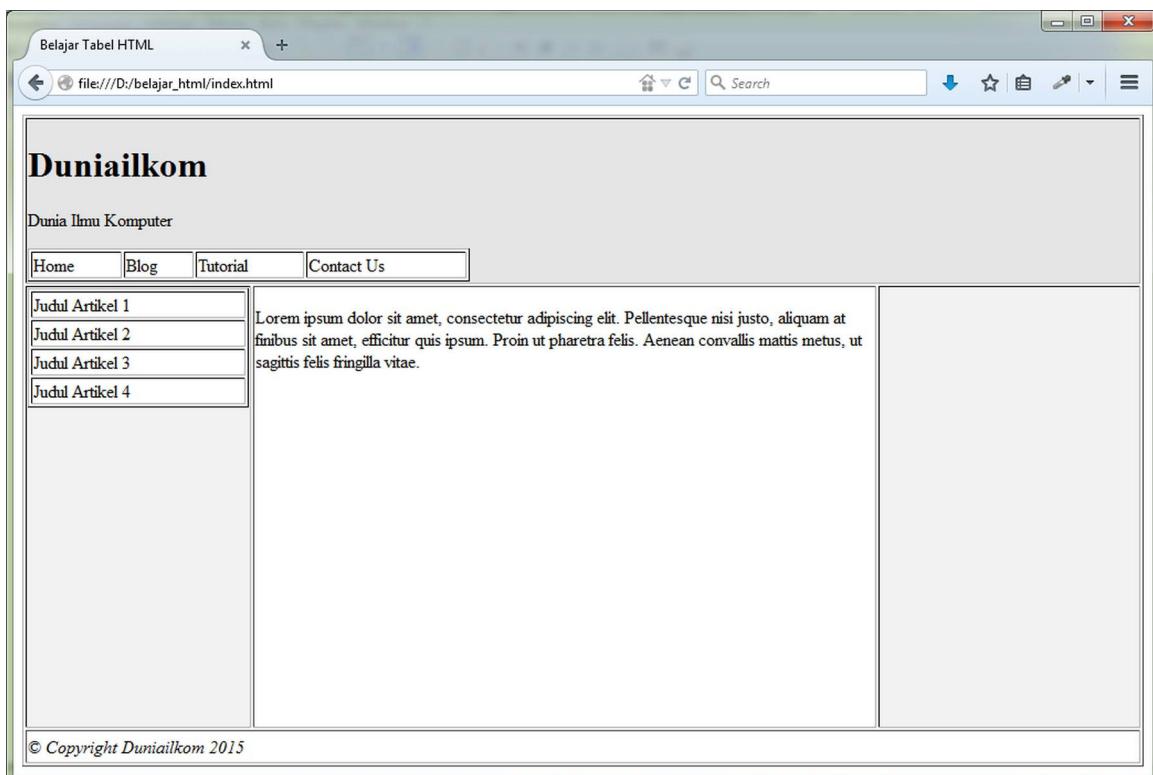
index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Tabel HTML</title>
6 </head>
7 <body>
8 <table border=1>
9   <tr height=150 bgcolor=#E5E5E5>
10    <td colspan=3 valign=bottom align=left>
11      <table border=1 width=400 bgcolor=white>
12        <h1>Duniaikom</h1><p>Dunia Ilmu Komputer</p>

```

```
13    <tr>
14        <td>Home</td><td>Blog</td>
15        <td>Tutorial</td><td>Contact Us</td>
16    </tr>
17    </table>
18    </td>
19 </tr>
20 <tr height=400>
21     <td width=200 bgcolor=#F2F2F2 valign=top>
22     <table border=1 width=200 bgcolor=white>
23         <tr><td>Judul Artikel 1</td></tr>
24         <tr><td>Judul Artikel 2</td></tr>
25         <tr><td>Judul Artikel 3</td></tr>
26         <tr><td>Judul Artikel 4</td></tr>
27     </table>
28     </td>
29     <td width=700 valign=top><p>Lorem ipsum dolor sit amet,
30     consectetur adipiscing elit. Pellentesque nisi justo,
31     aliquam at finibus sit amet, efficitur quis ipsum.
32     Proin ut pharetra felis. Aenean convallis mattis metus,
33     ut sagittis felis fringilla vitae.</p></td>
34     <td width=300 bgcolor=#F2F2F2></td>
35 </tr>
36 <tr height=30>
37     <td colspan=3>&copy; <i>Copyright DuniaIlkom 2015</i></td>
38 </tr>
39 </table>
40 </body>
41 </html>
```



Gambar: Tampilan layout halaman web dengan menggunakan tabel

Seperti yang terlihat, untuk mendapatkan hasil di atas, saya menggunakan berbagai atribut untuk menghasilkan tampilan tabel serta menggunakan tabel di dalam tabel. Struktur seperti ini membuat kode HTML susah diidentifikasi. Banyaknya tag-tag tabel yang dibutuhkan juga membuat penulisan kode menjadi tidak efisien.

Jika anda belum belajar CSS, tampilan layout di atas bisa digunakan untuk membuat website sederhana. Namun jika anda ‘serius’ ingin mempelajari web programming, saya sangat sarankan untuk meluangkan waktu dan tenaga mempelajari CSS. Dengan CSS, tampilan layout yang didapat akan jauh lebih baik dan mudah dikelola.

10.17 Penutup: Table Element

Pada bab kali ini kita telah mempelajari berbagai tag dan atribut yang digunakan untuk membuat tabel. Sebagai penutup saya akan mencontohkan pembuatan struktur tabel dengan HTML, dan mengubah tampilannya menggunakan CSS:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Belajar Table Element</title>
5   <style>
6     table {
7       border-collapse:collapse;
8       border-spacing:0;
9       border:1px black solid;
10      width:40%;
11    }
12    th, td {
13      padding:10px 20px;
14      border:1px black solid;}
15    th{
16      background-color: #87F97B;
17    }
18    tr:nth-child(even) {
19      background-color: #EDEDED;
20    }
21    tr:nth-child(odd) {
22      background-color: #FFF;
23    }
24  </style>
25 </head>
26 <body>
27 <h3>Tabel HTML</h3>
28 <table>
29 <caption>Tabel Simpanan Peserta</caption>
30   <thead>
31     <tr>
32       <th>No</th>
33       <th>Nama Peserta</th>
34       <th>Alamat</th>
35       <th>Simpanan</th>
36     </tr>
37   </thead>
38   <tfoot>
39     <tr>
40       <td colspan="3">Total</td>
41       <td>350.000</td>
42     </tr>
43   </tfoot>
44 <tbody>
```

```
45  <tr>
46    <td>1.</td>
47    <td>Andi Suryono</td>
48    <td>Jl. Kemerdekaan No.17</td>
49    <td>50.000</td>
50  </tr>
51  <tr>
52    <td>2.</td>
53    <td>Susi Handayani</td>
54    <td>Jl. Kebangsaan No.225</td>
55    <td>300.000</td>
56  </tr>
57  <tr>
58    <td>3.</td>
59    <td>Roy Pratama</td>
60    <td>Jl. Merdeka No.32</td>
61    <td>1.000.000</td>
62  </tr>
63  <tr>
64    <td>4.</td>
65    <td>Tia Suryani</td>
66    <td>Jl. Jelajah No.111</td>
67    <td>1.555.000</td>
68  </tr>
69  </tbody>
70 </table>
71 </body>
72 </html>
```

No	Nama Peserta	Alamat	Simpanan
1.	Andi Suryono	Jl. Kemerdekaan No.17	50.000
2.	Susi Handayani	Jl. Kebangsaan No.225	300.000
3.	Roy Pratama	Jl. Merdeka No.32	1.000.000
4.	Tia Suryani	Jl. Jelajah No.111	1.555.000
Total			350.000

Gambar: Contoh struktur tabel HTML dengan CSS

Terlepas dari kode CSS yang saya gunakan, struktur tabel di atas menggunakan sebagian besar tag-tag yang telah kita pelajari.

Tabel merupakan salah satu struktur penting untuk penyajian data. Jika anda berurusan dengan database, apa yang kita pelajari dalam bab ini menjadi sangat berguna untuk menyajikan hasil database ke dalam bentuk HTML.

Dalam bab selanjutnya kita akan membahas salah satu fitur paling penting di dalam sebuah website: Form. Pembahasan form akan dibagi menjadi 2 bab : Form Element, dan Form HTML5 Element.

11. Form Element

Form merupakan salah satu aspek terpenting di dalam sebuah website. Hampir setiap web memiliki form sebagai sarana interaksi dengan pengujung. Apakah itu digunakan untuk *register*, *login*, *polling*, atau untuk pencarian (*search box*).

Proses pembuatan form sebenarnya cukup kompleks, karena kita tidak hanya berurusan dengan HTML saja, tetapi juga dengan teknologi web lain seperti **PHP** (untuk memproses form), **MySQL** (untuk menyimpan hasil form), dan **JavaScript** (membuat validasi).

HTML menyediakan banyak tag dan atribut yang berkaitan dengan form. Ditambah lagi HTML5 membawa berbagai fitur baru seperti *validasi*. Oleh karena itu, pembahasan tentang form akan dipecah ke dalam 2 bab.

Karena pembahasan mengenai form cukup banyak, saya akan membaginya menjadi 2 bab. Dalam bab ini akan dibahas tentang cara penggunaan form dasar. Tag dan atribut ini sudah didukung penuh oleh semua web browser, termasuk web browser lama seperti IE 6, 7, dan 8.

Sedangkan dalam bab selanjutnya kita akan fokus membahas fitur baru yang dibawa oleh HTML5. Fitur-fitur ini hanya didukung oleh web browser modern.

11.1 Form Element

Seluruh objek form (*text box*, *radio button*, *checkbox*, dll) harus berada diantara tag pembuka `<form>` dan tag penutup `</form>`. Tag ini berguna sebagai *container* atau penampung dari form. Berapapun jumlah objek form yang berada diantara tag ini, masih dianggap sebagai satu form.

Jumlah form dalam tiap halaman HTML tidak dibatasi, sehingga kita bisa membuat 2, 3 atau 10 form dalam 1 halaman. Tag `<form>` termasuk ke dalam *block level element*.

Berikut contoh penulisan kerangka sebuah form:

```
<form>
...
...
...
</form>
```



Dalam buku ini saya menggunakan istilah ‘**objek form**’ untuk menyebut bagian-bagian form seperti kotak inputan (*text box*), tombol radio (*radio button*), *checkbox*, tombol *submit*, dll. Total, HTML menyediakan belasan objek form yang bisa kita gunakan (belum termasuk objek form dari HTML5).

Atribut Action

Atribut **action** digunakan untuk menulis alamat file yang menjadi tujuan pengiriman data. Biasanya nilai dari atribut **action** berupa alamat sebuah halaman PHP (atau bahasa pemrograman server lain) yang digunakan untuk memproses isi form.

Berikut contoh penulisan atribut **action** dalam form:

```
<form action="prosesform.php" >  
...isi form...  
</form>
```

Form di atas akan mengirim hasil isian form ke halaman **prosesform.php** untuk selanjutnya diproses di sisi server.

Atribut Method

Atribut **method** berfungsi untuk memberitahu web browser bagaimana cara pengiriman data form. Atribut ini dapat diisi dengan nilai **get** atau **post**.

Perbedaan antara **get** dan **post** lebih bersifat teknis. Sebagai gambaran, jika kita menulis atribut **method="get"**, maka isian data form akan ditambahkan ke dalam alamat URL (terlihat pada *address bar* web browser). Sedangkan jika menggunakan **method="post"**, isi data form ini tidak akan terlihat.

Sesuai dengan namanya, **method="get"** digunakan jika form akan ‘mengambil’ sesuatu dari server (seperti menampilkan artikel atau menampilkan hasil pencarian), sedangkan **method="post"** lebih cocok digunakan untuk form yang akan ‘menginput’ sesuatu ke dalam web server (seperti form *registrasi*) atau form dengan data yang bersifat rahasia (seperti form *login*). Walaupun demikian, sifatnya hanya sebagai rekomendasi. Kita bisa menggunakan **method get** maupun **post** untuk tujuan apa saja.

Baik **method get** maupun **post** memiliki batas maksimal karakter yang dapat dikirim. Ini bergantung ke pada web browser dan web server yang digunakan. Sebagai gambaran, **method get** umumnya mendukung hingga 2kB karakter (sekitar 2000 karakter), sedangkan **method post** bisa mendukung hingga 2GB karakter (sekitar 2 miliar karakter). Oleh karena itu, jika form yang dikirim cukup panjang, sebaiknya menggunakan **method post**.



Jika atribut **method** tidak di tulis, web browser akan menggunakan **method="get"** sebagai nilai default.

Berikut contoh penulisan atribut **method**:

```
<form action="prosesform.php" method="post">
    ...isi form...
</form>
```

Form di atas akan mengirimkan nilai data form ke halaman prosesform.php dengan menggunakan method post.

11.2 Input Element Type Text (text box)

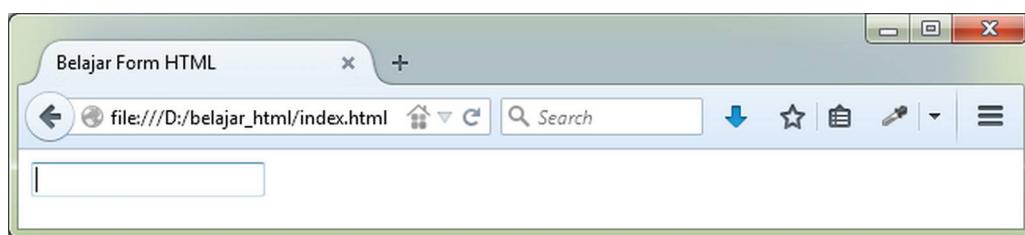
Objek form yang paling sederhana (dan juga paling banyak digunakan) adalah **text box**. *Text box* terlihat sebagai kotak persegi panjang dimana pengguna bisa mengetik sesuatu ke dalam kotak ini. Text box disebut juga dengan ‘*input type text*’, karena untuk membuat text box, kita menambahkan atribut *type="text"* pada tag *<input>*.

Tag *<input>* merupakan tag ‘universal’ yang digunakan untuk membuat hampir seluruh objek form. Nilai dari atribut **type** inilah yang akan membedakan jenis objek form yang satu dengan yang lain. Penulisan tag *<input>* juga tidak memerlukan tag penutup atau dikenal sebagai *self closing tag*.

Berikut adalah contoh pembuatan text box di dalam HTML:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="UTF-8">
5     <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="post">
9     <input type="text">
10 </form>
11 </body>
12 </html>
```



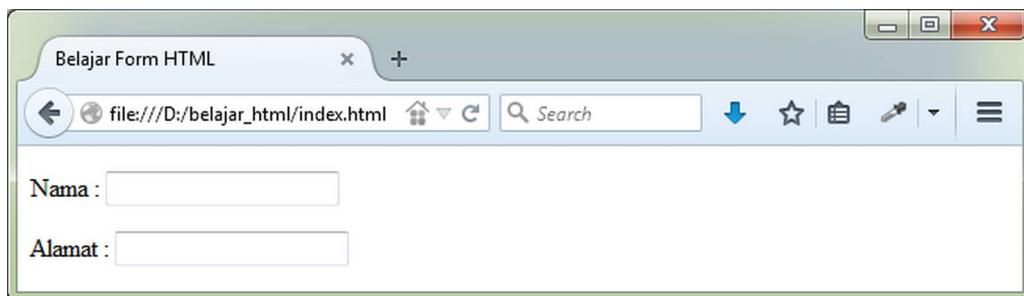
Gambar: Tampilan form `input type="text"`

Jika anda menjalankan kode di atas, akan terlihat sebuah kotak inputan.

Kita juga bisa menambahkan sedikit teks sebelum tag *<input>* sebagai penjelasan mengenai kegunaan text box ini, seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="post">
9   <p>Nama : <input type="text"></p>
10  <p>Alamat : <input type="text"></p>
11 </form>
12 </body>
13 </html>
```



Gambar: Tampilan form `input type="text"` dengan teks keterangan

Dalam contoh di atas, saya membuat 2 buah text box untuk menampung nilai **nama** dan **alamat**.



Text box di atas berada di dalam tag `<p>`. Gaya penulisan seperti ini sesuai dengan spesifikasi HTML5 dimana setiap objek form merupakan bagian yang saling terpisah dan dianggap sebagai paragraf. Walaupun begitu, anda tidak harus mengikuti penulisan seperti ini. Beberapa programmer web memilih menggunakan tag `` atau `<div>` untuk setiap objek form.

Atribut Name

Tag `<input>` dapat ditambah dengan berbagai atribut. Salah satu yang paling penting adalah atribut **name**. Atribut ini berperan sebagai penanda dan berguna pada saat form akan diproses.

Berikut adalah contoh penggunaan atribut **name** untuk text box:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="post">
9   <p>Nama : <input type="text" name="nama_user"></p>
10  <p>Alamat : <input type="text" name="alamat_user"></p>
11 </form>
12 </body>
13 </html>
```

Jika anda menjalankan kode HTML di atas, tidak akan terlihat perubahan apa-apa. Atribut **name** digunakan hanya untuk pemrosesan form dan tidak ada pengaruh dengan tampilan. Atribut **name** bisa digunakan untuk semua objek form.

Untuk menggambarkan fungsi dari atribut **name**, form di atas dapat diakses dari bahasa pemrograman PHP menggunakan kode berikut:

```
<?php
echo $_POST['nama_user'];
echo $_POST['alamat_user'];
?>
```



Pembahasan mengenai PHP tidak saya bahas dalam buku ini. Kode di atas hanya untuk menjelaskan kegunaan atribut **name**. Perhatikan bahwa saya menggunakan `$_POST` karena di dalam `<form>` menggunakan `method="post"`. Jika saya mengganti atribut `method` menjadi "get", maka di dalam PHP menggunakan variabel `$_GET`.

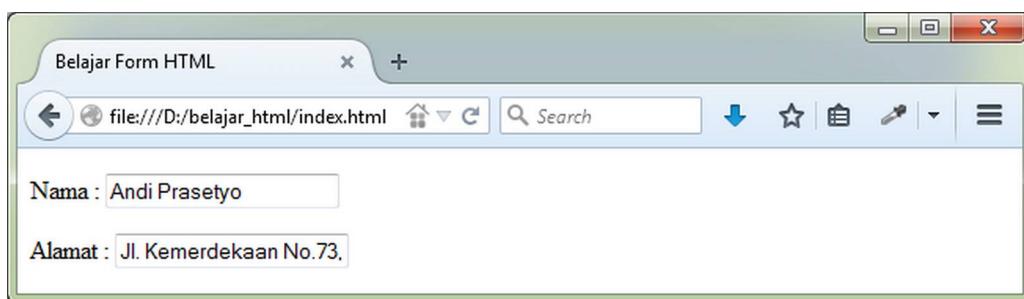
Atribut Value

Atribut **value** berguna untuk memberi nilai pada sebuah objek form. Khusus untuk **text box**, nilai dari atribut **value** akan ditampilkan sebagai nilai awal. Apabila pengguna tidak mengubahnya, nilai inilah yang akan dikirim ke server untuk diproses.

Berikut contoh penggunaan atribut **value** pada **text box**:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="post">
9   <p>Nama : <input type="text" name="nama_user" value="Andi Prasetyo"></p>
10  <p>Alamat : <input type="text" name="alamat_user"
11    value="Jl. Kemerdekaan No.73, Palembang"></p>
12 </form>
13 </body>
14 </html>
```



Gambar: Tampilan form `input type="text"` dengan atribut `value`

Ketika anda menjalankan kode di atas, text box otomatis berisi nilai sesuai dengan atribut `value`. Atribut ini bisa digunakan untuk seluruh objek form lainnya.



Apabila anda ingin teks menghilang pada saat mulai mengetik, HTML5 menyediakan atribut `placeholder` (yang akan kita bahas dalam bab berikutnya).

Atribut Size

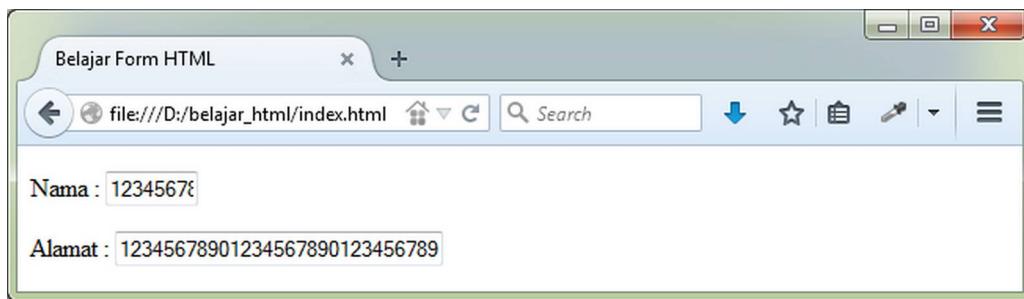
Kita juga bisa mengatur panjang **text box** yang ditampilkan dengan menggunakan atribut `size`. Nilai dari atribut ini berupa angka yang dinyatakan dalam satuan huruf.

Sebagai contoh, jika saya membuat `size="5"`, maka panjang text box akan di-set sepanjang 5 karakter. Akan tetapi karena panjang satu karakter berlainan dengan karakter lain dan juga dipengaruhi jenis font, nilai ini hanya sebagai acuan.

Berikut contoh penggunaan atribut `size`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="post">
9   <p>Nama : <input type="text" name="nama_user" size="5"></p>
10  <p>Alamat : <input type="text" name="alamat_user" size="30"></p>
11 </form>
12 </body>
13 </html>
```



Gambar: Tampilan form `input type="text"` dengan atribut `size`

Seperti yang terlihat dari tampilan form, walaupun menggunakan atribut `size="5"`, saya bisa menginput sampai 7 angka. Apabila atribut ini tidak ditulis, nilai default yang digunakan web browser adalah adalah `size="20"`. Atribut ini bisa digunakan untuk seluruh objek form yang berbentuk text box.

Atribut Readonly

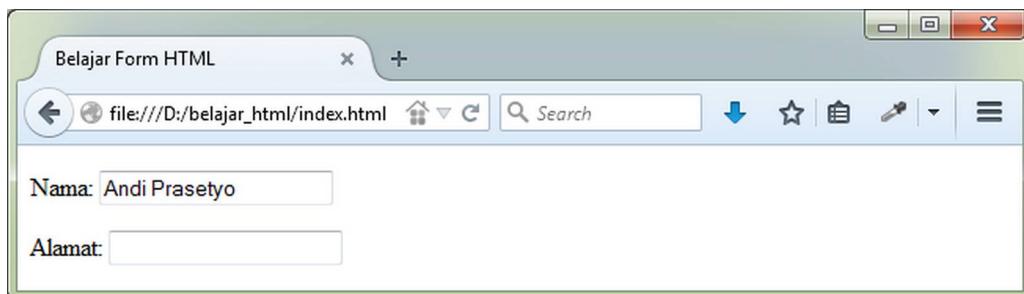
Jika kita ingin sebuah objek form tidak dapat diubah nilainya oleh pengguna, bisa menggunakan atribut **readonly**.

Atribut **readonly** akan membuat objek form tidak bisa diubah nilainya, namun pengguna masih bisa berinteraksi dengan objek ini, misalnya men-klik teks. Tampilan yang dihasilkan juga tampak seperti objek form ‘biasa’.

Berikut contohnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="post">
9   <p>
10    Nama : <input type="text" name="nama_user" value="Andi Prasetyo" readonly>
11   </p>
12   <p>
13    Alamat : <input type="text" name="alamat_user" readonly>
14   </p>
15 </form>
16 </body>
17 </html>
```



Gambar: Tampilan form `input type="text"` dengan atribut `readonly`

Kedua text box di atas tidak akan bisa diisi. Namun seperti yang terlihat, keduanya tampak seperti objek form ‘normal’. Atribut ini bisa digunakan untuk mayoritas objek form lainnya.



Penulisan atribut `readonly` seperti contoh di atas merupakan cara yang digunakan oleh HTML. Jika anda pernah mempelajari XHTML, maka cara penulisannya adalah `readonly="readonly"`. Anda juga bisa menggunakan cara penulisan seperti ini.

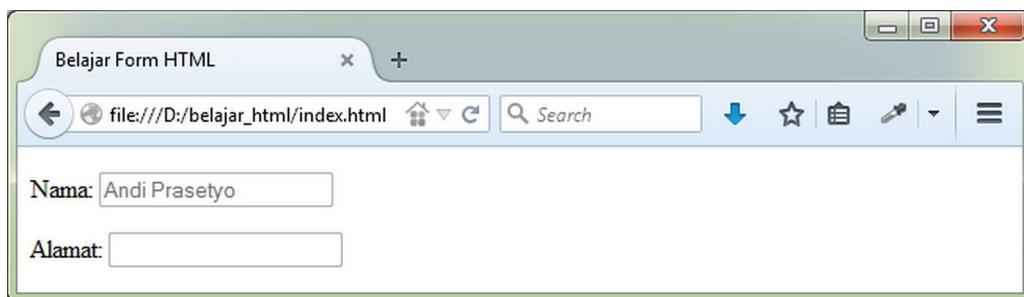
Atribut Disabled

Efek atribut **disabled** mirip dengan **readonly**, yakni membuat objek form tidak bisa diakses. Akan tetapi, berbeda dengan `readonly`, tampilan objek form **disabled** akan berwarna abu-abu sebagai tanda tidak bisa diakses.

Berikut adalah contoh penggunaan atribut **disabled**:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="post">
9   <p>
10    Nama : <input type="text" name="nama_user" value="Andi Prasetyo" disabled>
11   </p>
12   <p>
13    Alamat : <input type="text" name="alamat_user" disabled>
14   </p>
15 </form>
16 </body>
17 </html>
```



Gambar: Tampilan form `input type="text"` dengan atribut `disabled`

Efek `readonly` dan `disabled` umumnya digunakan bersamaan dengan JavaScript. Dengan demikian, kita bisa mengaktifkan kembali objek form ini menggunakan JavaScript.



Sama seperti `readonly`, kita juga bisa menulis atribut ini dengan gaya penulisan XHTML, yakni `disabled="disabled"`.

Atribut **maxlength**

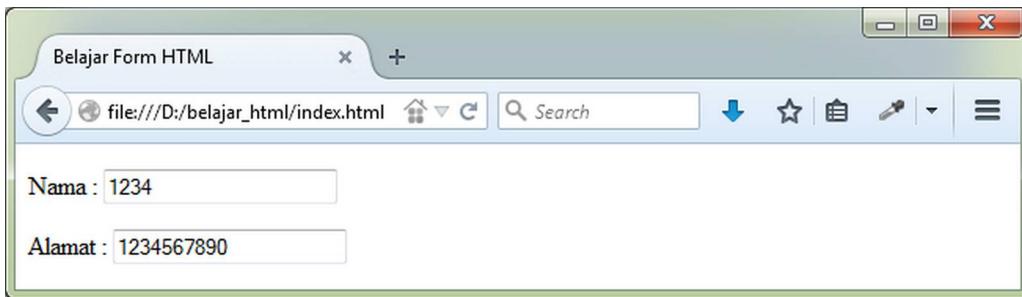
Atribut **maxlength** digunakan untuk membatasi jumlah digit karakter pada sebuah text box. Nilai dari atribut ini berupa angka yang menunjukkan seberapa banyak maksimal digit karakter yang dibolehkan.

Sebagai contoh, jika saya membuat `maxlength="10"`, maka text box tersebut akan membatasi input sebanyak 10 digit karakter.

Berikut contoh penggunaan atribut **maxlength**:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="post">
9   <p>Nama : <input type="text" name="nama_user" maxlength="4"></p>
10  <p>Alamat : <input type="text" name="alamat_user" maxlength="10"></p>
11 </form>
12 </body>
13 </html>
```



Gambar: Tampilan form `input type="text"` dengan atribut `maxlength`

Jika anda menjalankan kode di atas, kita tidak bisa menginput lebih dari 4 karakter sebagai **nama_user**.

Namun perlu diingat bahwa pembatasan ini dapat dihapus oleh pengguna. Agar lebih aman, sebaiknya kita memeriksa kembali panjang karakter di sisi server (menggunakan PHP).

11.3 Input Element Type Submit

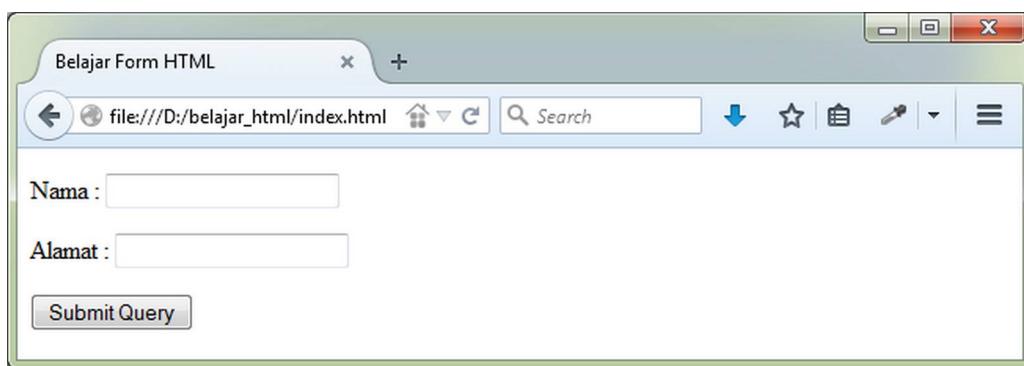
Tag `<input type="text">` yang kita pelajari sebelum ini merupakan salah satu dari banyak objek form di dalam HTML. Sebelum kita mempelajari objek-objek form lain, saya akan membahas terlebih dahulu cara mengirimkan form ke server untuk diproses dengan menggunakan tombol **submit**.

Untuk mengirimkan form ke server, HTML menyediakan tag `<input type="submit">`. Tag ini masih menggunakan tag `<input>`, akan tetapi tampilannya berbentuk tombol (*button*). Berikut contoh penggunaannya:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="nama_user"></p>
10  <p>Alamat : <input type="text" name="alamat_user"></p>
11  <p><input type="submit"></p>
12 </form>
13 </body>
14 </html>
```



Gambar: Tampilan form `input type="submit"`

Seperti yang terlihat, tag `<input type="submit">` berbentuk tombol. Ketika tombol ini di klik, form akan langsung dikirim ke halaman `prosesform.php` (sesuai dengan atribut `action` pada tag `form`).

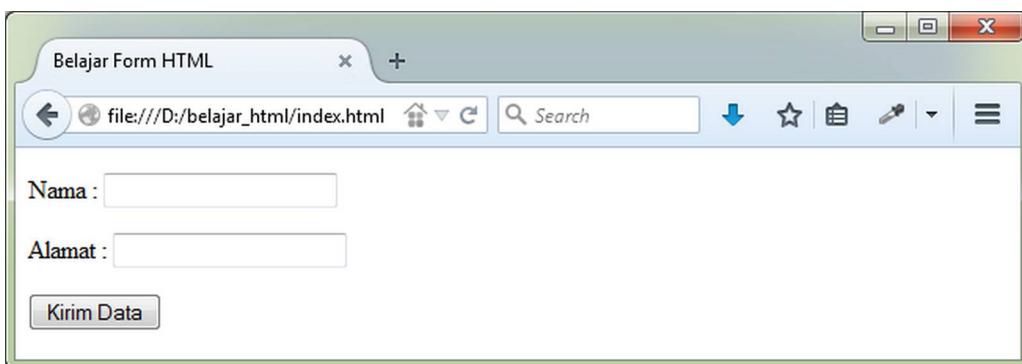
Secara default, tombol submit akan ditampilkan beragam. Mozilla Firefox akan menampilkan tombol ini dengan nama “*Submit Query*” sedangkan Google Chrome akan menampilkan “*Submit*” saja. Untuk mengubah text ini, kita bisa menggunakan atribut `value`. Berikut contohnya:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="nama_user"></p>
```

```
10 <p>Alamat : <input type="text" name="alamat_user"></p>
11 <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>
```



Gambar: Tampilan form `input type="submit"` dengan atribut `value`

Kali ini tombol submit akan ditampilkan dengan teks : **Kirim Data**.

Jika diperhatikan, dalam contoh kode di atas saya menggunakan atribut `method="get"`, sehingga hasil form dapat terlihat pada alamat URL web browser. Apabila saya mengisi form `nama=Anto`, dan `alamat=Padang`, dan men-klik tombol “**Kirim Data**”, maka pada bagian address bar akan terlihat kode berikut:

file:///D:/belajar_html/prosesform.php?nama_user=Anto&alamat_user=Padang

Perhatikan karakter setelah tanda tanya, yakni bagian:

`nama_user=Anto&alamat_user=Padang`

Inilah hasil form yang akan dikirim kepada web server (PHP) untuk diproses. Tanda “**&**” digunakan untuk memisahkan nilai objek form dengan nilai objek form lainnya. Apabila anda mengganti atribut method form menjadi **post**, maka pada alamat URL tidak akan terlihat apa-apa (namun nilai form tetap terkirim).

Walaupun tampak tidak relevan, kita juga bisa menambahkan atribut **name** pada tombol submit, sehingga nilainya ikut dikirim ke server. Hal ini berfungsi untuk proses konfirmasi form (menggunakan PHP). Berikut contohnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="nama_user"></p>
10  <p>Alamat : <input type="text" name="alamat_user"></p>
11  <p><input type="submit" value="Kirim Data" name="form_register"></p>
12 </form>
13 </body>
14 </html>
```

Atribut Target

Pada saat form dikirim, secara otomatis halaman web akan berpindah ke halaman baru sesuai dengan nilai dari atribut **action**. Untuk mengatur apakah halaman ini ditampilkan pada jendela yang sama atau pada jendela / tab baru, kita bisa menambahkan atribut **target** pada tag `<form>`.

Nilai untuk atribut **target** ini sama dengan yang digunakan pada tag `<a>`, yaitu salah satu dari `_self`, `_blank`, `_parent`, `_top`, atau `framename`. Penjelasan mengenai fungsi dari masing-masing nilai telah kita bahas pada bab 6 ketika membahas tag `<a>`.

Sebagai contoh, kode form berikut ini akan menampilkan halaman `prosesform.php` pada tab baru:

index.html

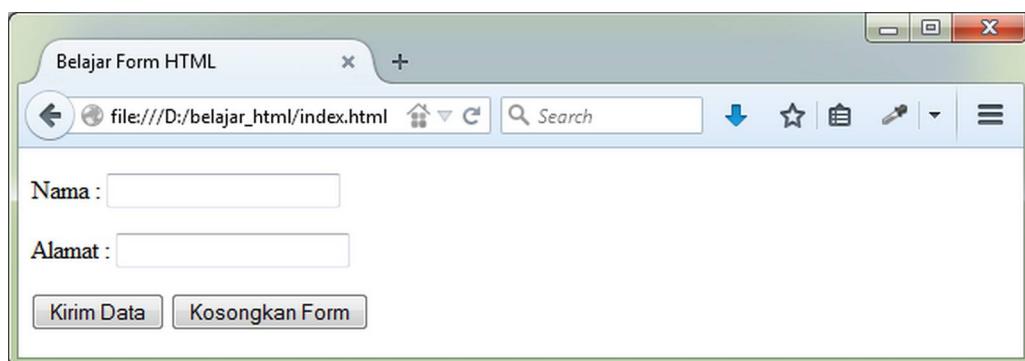
```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get" target="_blank">
9   <p>Nama : <input type="text" name="nama_user"></p>
10  <p>Alamat : <input type="text" name="alamat_user"></p>
11  <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>
```

11.4 Input Element Type Reset

HTML juga menyediakan tombol **reset** untuk mengosongkan isi form. Penggunaannya mirip dengan tombol **submit**, namun kali ini kita menggunakan atribut `type="reset"`. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="nama_user"></p>
10  <p>Alamat : <input type="text" name="alamat_user"></p>
11  <p>
12    <input type="submit" value="Kirim Data">
13    <input type="reset" value="Kosongkan Form">
14  </p>
15 </form>
16 </body>
17 </html>
```



Gambar: Tampilan form `input type="reset"`



Tombol reset tidak terlalu sering digunakan, terutama untuk menghindari pengunjung yang tidak sengaja men-klik tombol reset ketika selesai mengisi form yang panjang. Untuk mengosongkan form tanpa tombol reset, cukup dengan membuka ulang halaman form.

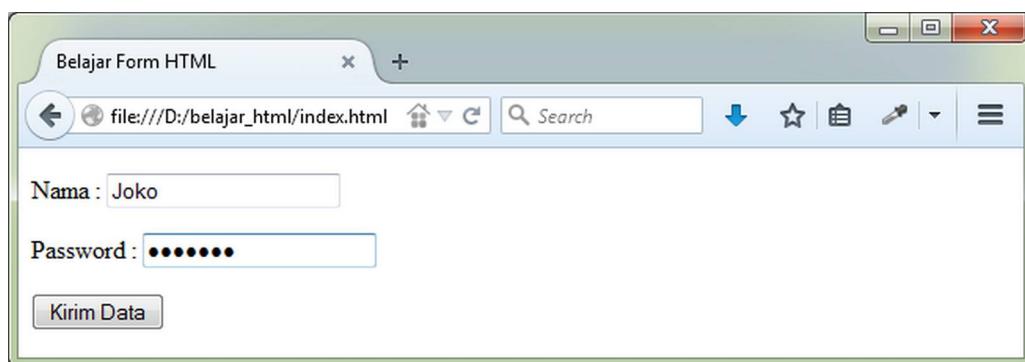
11.5 Input Element Type Password

Khusus untuk penginputan password, HTML menyediakan objek form khusus, yakni **input type="password"**. Untuk menggunakannya, kita menambahkan atribut `type="password"` pada tag `<input>`.

Berikut contoh penggunaan tag input type password:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="nama_user"></p>
10  <p>Password : <input type="password" name="password_user"></p>
11  <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>
```



Gambar: Tampilan form `input type="password"`

Tampilan text box password tidak berbeda dengan text box reguler. Akan tetapi, ketika kita mulai mengetik, karakter yang diinput akan 'disamarkan'.

Walaupun demikian, ketika kita men-klik tombol submit, nilainya tetap terlihat dalam URL (apabila menggunakan `method="get"`). Oleh karena itu, untuk informasi sensitif seperti password sebaiknya menggunakan `method="post"`.

Seluruh atribut yang berlaku pada text box reguler juga bisa digunakan untuk text box password, seperti `size`, `maxlength`, `readonly`, `disabled`, dll.

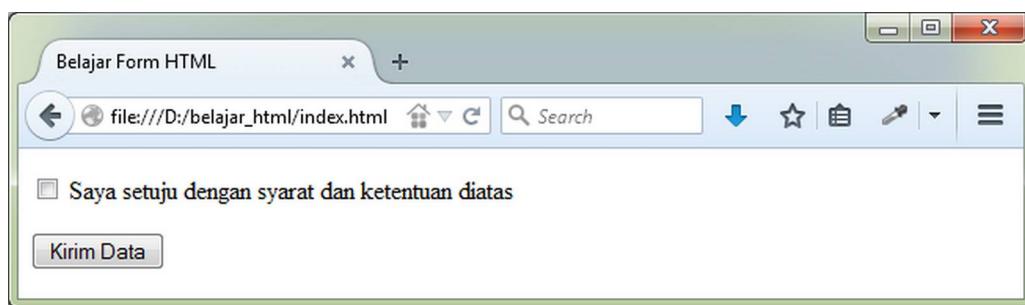
11.6 Input Element Type Checkbox

Checkbox adalah objek form yang tampilannya berupa kotak persegi dimana pengguna bisa men-klik (ceklist) kotak isian untuk memilih. Untuk membuat **checkbox** kita menggunakan atribut `type="checkbox"` pada tag `<input>`.

Berikut contoh penggunaan tag input `type="checkbox"`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p><input type="checkbox" name="persetujuan" value="setuju">
10  Saya setuju dengan syarat dan ketentuan di atas</p>
11  <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>
```



Gambar: Tampilan form `input type="checkbox"`

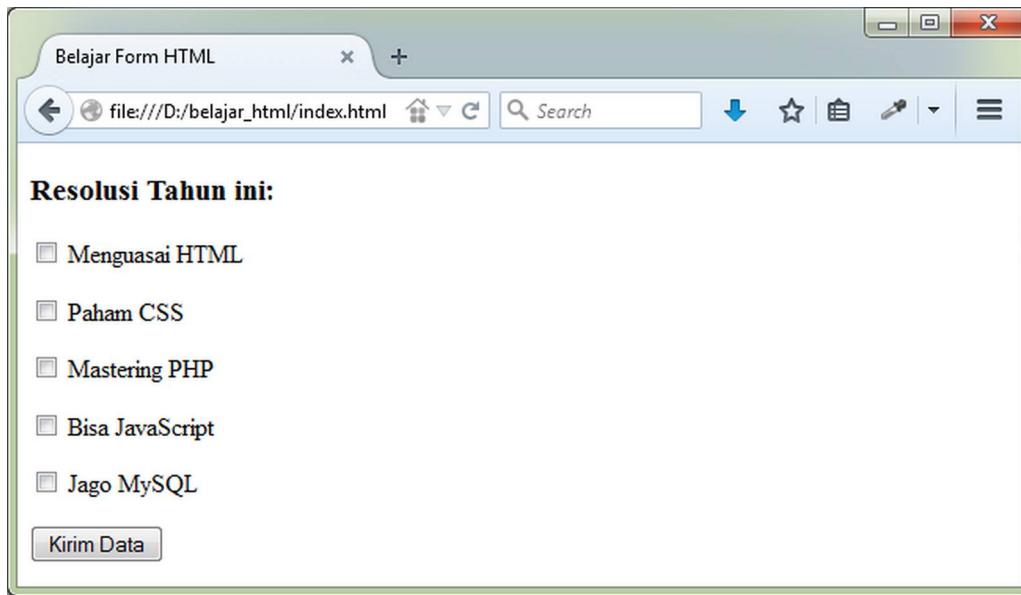
Dalam contoh di atas, saya membuat sebuah checkbox dengan atribut `name="persetujuan"` dan `value="setuju"`. Ketika pengguna memilih pilihan di atas dan men-submit form, hasil yang terlihat pada address bar adalah sebagai berikut:

file:///D:/belajar_html/prosesform.php?persetujuan=setuju

Objek form checkbox cocok digunakan untuk inputan dimana pengguna bisa memilih lebih dari 1 pilihan, seperti contoh berikut ini:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <h3>Resolusi Tahun ini:</h3>
9 <form action="prosesform.php" method="get">
10   <p><input type="checkbox" name="target1" value="HTML"> Menguasai HTML</p>
11   <p><input type="checkbox" name="target2" value="CSS"> Paham CSS</p>
12   <p><input type="checkbox" name="target3" value="PHP"> Mastering PHP</p>
13   <p><input type="checkbox" name="target4" value="CSS"> Bisa JavaScript</p>
14   <p><input type="checkbox" name="target5" value="PHP"> Jago MySQL</p>
15   <p><input type="submit" value="Kirim Data"></p>
16 </form>
17 </body>
18 </html>
```



Gambar: Tampilan form `input type="checkbox"`

Pada kode di atas, saya menggunakan atribut **name** dan **value** yang berlainan, sehingga setiap checkbox merupakan objek form yang saling terpisah. Anda bisa mencoba memilih seluruh pilihan yang ada, submit form, dan pelajari hasil yang dikirim pada bagian URL.

Atribut Checked

HTML menyediakan atribut **checked** yang berfungsi agar kotak checkbox langsung terpilih pada saat halaman ditampilkan. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <h3>Resolusi Tahun ini:</h3>
9 <form action="prosesform.php" method="get">
10  <p><input type="checkbox" name="target1" value="HTML" checked>
11    Menguasai HTML</p>
12  <p><input type="checkbox" name="target2" value="CSS"> Paham CSS</p>
13  <p><input type="checkbox" name="target3" value="PHP"> Mastering PHP</p>
14  <p><input type="checkbox" name="target4" value="CSS" checked>
15    Bisa JavaScript</p>
16  <p><input type="checkbox" name="target5" value="PHP"> Jago MySQL</p>
17  <p><input type="submit" value="Kirim Data"></p>
18 </form>
19 </body>
20 </html>
```

Ketika kode di atas dijalankan, pilihan checkbox *Menguasai HTML* dan *Bisa JavaScript* akan langsung terpilih.,



Kita juga bisa menulis atribut **checked** dengan `checked="checked"`. Ini adalah cara penulisan XHTML.

11.7 Input Element Type Radio (radio button)

Jika kita ingin membatasi pilihan dimana pengguna hanya bisa memilih salah satu nilai saja, bisa menggunakan **radio button**. Contoh penggunaannya adalah untuk inputan jenis kelamin di dalam form. Pengguna hanya bisa memilih salah satu antara laki-laki atau perempuan, tetapi tidak keduanya :)

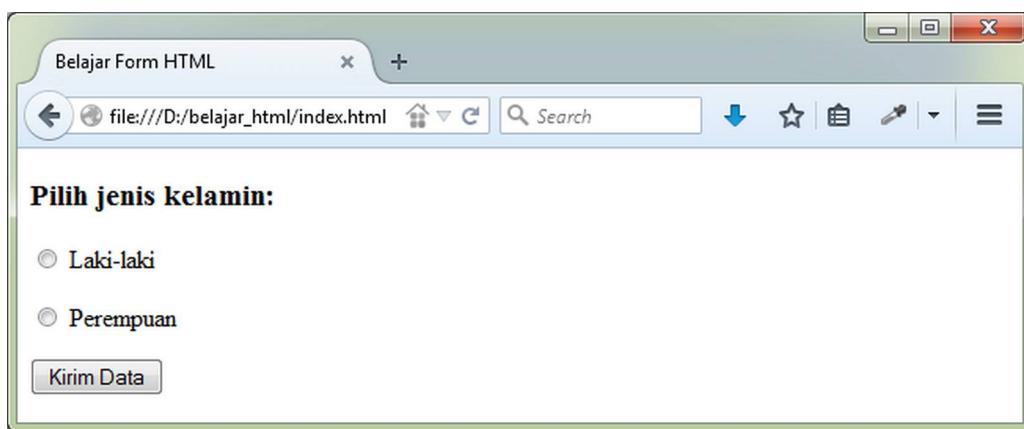
Untuk membuat radio button, kita menggunakan atribut `type="radio"` pada tag `<input>`. Berikut contoh penggunaannya:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <h3>Pilih jenis kelamin:</h3>
9 <form action="prosesform.php" method="get">
10  <p><input type="radio" name="jenis_kelamin" value="laki_laki"> Laki-laki</p>
11  <p><input type="radio" name="jenis_kelamin" value="perempuan"> Perempuan</p>
12  <p><input type="submit" value="Kirim Data"></p>
13 </form>
14 </body>
15 </html>

```



Gambar: Tampilan form `input type="radio"`

Dalam kode di atas, kita hanya bisa memilih salah satu dari pilihan yang ada. Penulisan radio button mirip dengan **check box**, namun perhatikan atribut **name**. Agar setiap radio button saling terhubung dengan radio button pasangannya atribut **name** harus diisi dengan nilai yang sama. Berikut adalah contoh lain dari radio button:

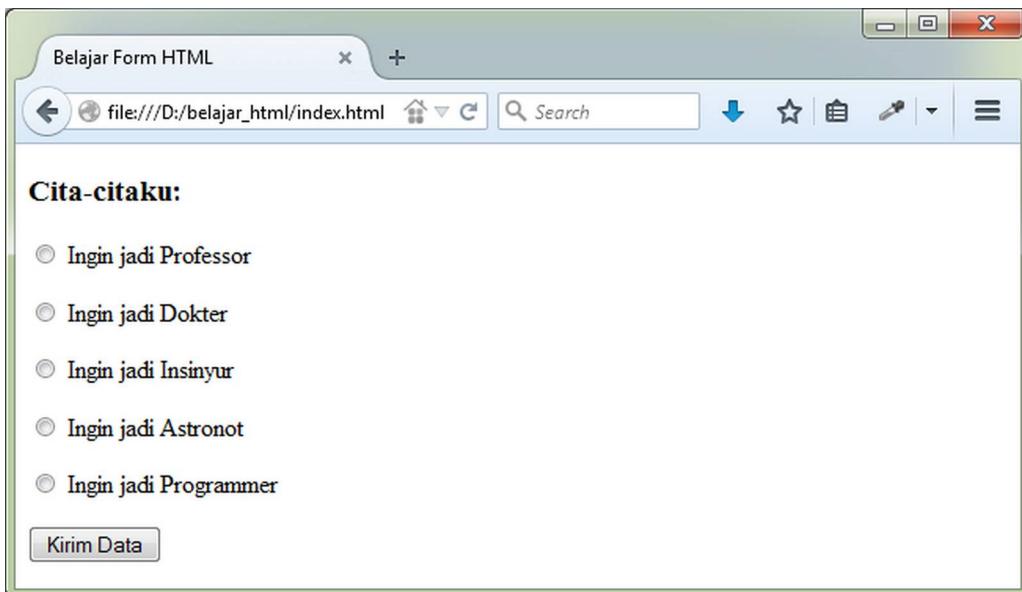
index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <h3>Cita-citaku:</h3>

```

```
9 <form action="prosesform.php" method="get">
10  <p><input type="radio" name="cita2" value="professor">
11  Ingin jadi Professor</p>
12  <p><input type="radio" name="cita2" value="dokter">
13  Ingin jadi Dokter</p>
14  <p><input type="radio" name="cita2" value="insinyur">
15  Ingin jadi Insinyur</p>
16  <p><input type="radio" name="cita2" value="astronot">
17  Ingin jadi Astronot</p>
18  <p><input type="radio" name="cita2" value="programmer">
19  Ingin jadi Programmer</p>
20  <p><input type="submit" value="Kirim Data"></p>
21 </form>
22 </body>
23 </html>
```



Gambar: Tampilan form `input type="radio"`

Kali ini pengguna hanya bisa memilih 1 jenis cita-cita saja. Jika kita ingin agar pengguna bisa memilih lebih dari 1 pilihan, bisa menggunakan check box.

Sama seperti objek form yang lain, atribut `value` digunakan sebagai nilai yang akan dikirim untuk diproses.

Kita juga bisa menggunakan atribut `checked` untuk menandai pilihan awal ketika form pertama kali ditampilkan. Karena sifat **radio button** yang hanya bisa dipilih satu pilihan saja, maka hanya 1 radio button yang bisa menggunakan atribut ini. Berikut contohnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <h3>Cita-citaku:</h3>
9 <form action="prosesform.php" method="get">
10  <p><input type="radio" name="cita2" value="professor">
11    jadi Professor</p>
12  <p><input type="radio" name="cita2" value="dokter">
13    Ingin jadi Dokter</p>
14  <p><input type="radio" name="cita2" value="insinyur">
15    Ingin jadi Insinyur</p>
16  <p><input type="radio" name="cita2" value="astronot">
17    Ingin jadi Astronot</p>
18  <p><input type="radio" name="cita2" value="programmer" checked>
19    Ingin jadi Programmer</p>
20  <p><input type="submit" value="Kirim Data"></p>
21 </form>
22 </body>
23 </html>
```

Kali ini ketika halaman di tampilkan pertama kali, pilihan **Ingin jadi Programmer** akan langsung terpilih.

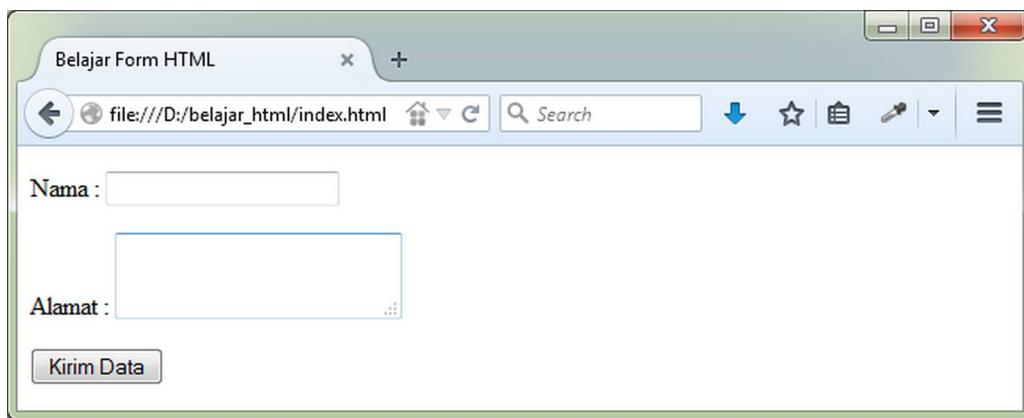
11.8 Textarea Element

Jika dilihat dari tampilannya, **textarea** mirip dengan **text box**, tetapi text area bisa menampung lebih banyak teks. Untuk membuat text area, kita menggunakan tag pembuka `<textarea>` dan tag penutup `</textarea>`.

Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="nama"></p>
10  <p>Alamat : <textarea name="alamat"></textarea></p>
11  <input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>
```



Gambar: Tampilan form **textarea**

Objek form **textarea** cocok digunakan untuk inputan data yang panjang, seperti alamat atau untuk bagian komentar pada sebuah web. **Textarea** juga memiliki berbagai atribut yang bisa digunakan untuk mengatur tampilan.

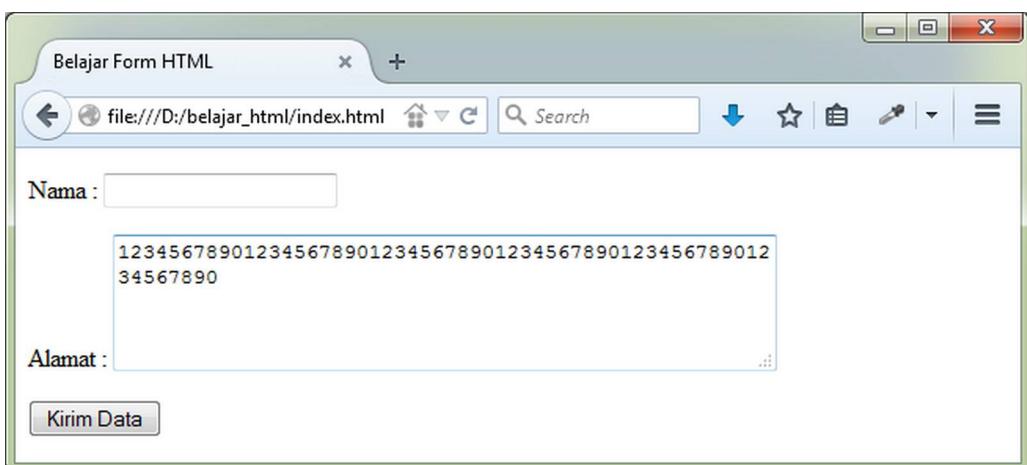
Atribut Rows dan Cols

Atribut **rows** dan **cols** digunakan untuk mengatur jumlah baris (**rows**) dan kolom (**cols**) dari **textarea**. Nilai kedua atribut ini berupa angka yang menunjukkan jumlah karakter untuk tinggi dan lebar **textarea**. Berikut contoh penggunaannya:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="nama"></p>
10  <p>Alamat : <textarea name="alamat" rows="4" cols="50"></textarea></p>
11  <input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>
```



Gambar: Tampilan form **textarea** dengan atribut **rows** dan **cols**

Saya membuat textarea dengan atribut `rows="4"` dan `cols="50"`, dengan demikian, web browser akan menampilkan textarea dengan tinggi 4 karakter, dan lebar 50 karakter. Namun seperti yang terlihat, nilai ini lebih tepatnya ‘rata-rata’ jumlah karakter. Pada tampilan di atas, saya bisa menginput sekitar 53 karakter angka pada baris pertama (dimana seharusnya hanya sebanyak 50 karakter).

Membuat Nilai Awal Textarea

Sedikit berbeda dengan kebanyakan objek form lain, untuk membuat nilai awal dari **textarea** kita tidak menggunakan atribut **value**, tetapi tinggal menyisipkan teks diantara tag pembuka dan tag penutup `<textarea>`. Berikut contohnya:

index.html

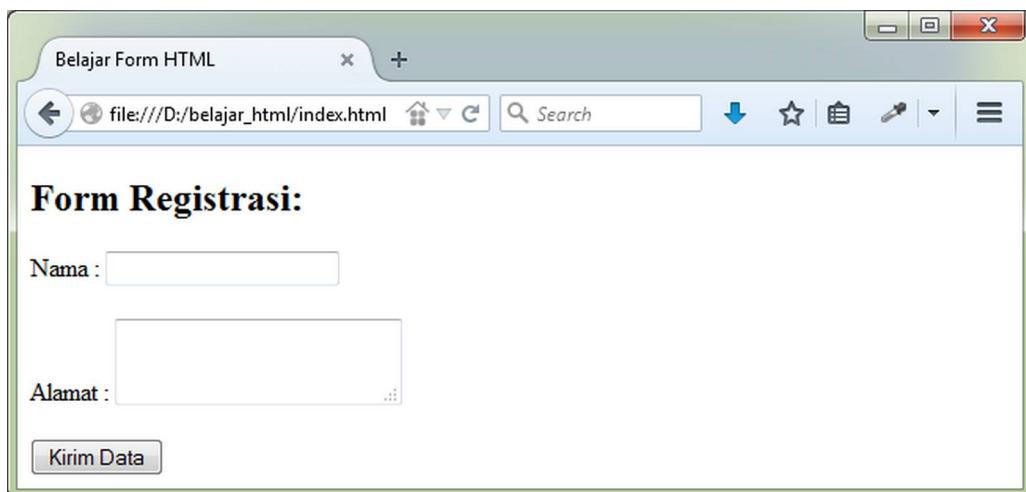
```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="nama"></p>
10  <p>Alamat : <textarea name="alamat" rows="4" cols="50">
11    Jl. Nangka no 45, Pekanbaru</textarea></p>
12  <input type="submit" value="Kirim Data"></p>
13 </form>
14 </body>
15 </html>
```

Atribut Maxlength

Kita juga bisa membatasi karakter yang bisa diinput pada textarea dengan menggunakan atribut **maxlength**. Atribut ini diisi dengan nilai angka yang menunjukkan berapa banyak jumlah maksimal karakter yang diperbolehkan. Seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <h2>Form Registrasi:</h2>
9 <form action="prosesform.php" method="get">
10  Nama : <input type="text" name="nama" >
11  <br><br>
12  Alamat : <textarea name="alamat" maxlength="10"></textarea>
13  <br><br>
14 <input type="submit" value="Kirim Data">
15 </form>
16 </body>
17 </html>
```



Gambar: Tampilan form `textarea` dengan atribut `maxlength`

Dalam contoh di atas, saya men-set nilai `maxlength` sebanyak 10 karakter, dengan demikian `textarea` tidak bisa diinput lebih dari 10 karakter. Namun perlu dicatat bahwa seseorang bisa mengubah batasan ini. Kita sebaiknya men-cek ulang disisi server (menggunakan PHP).

11.9 Select Element dan Option Element (menu dropdown)

Menu **dropdown** adalah sebuah objek form dimana pengguna bisa memilih dari pilihan yang telah tersedia. Untuk membuatnya, kita menggunakan 2 buah element, yakni tag `<select>` sebagai *container*, dan tag `<option>` untuk membuat pilihan.

Agar lebih mudah dipahami, berikut contoh penggunaannya:

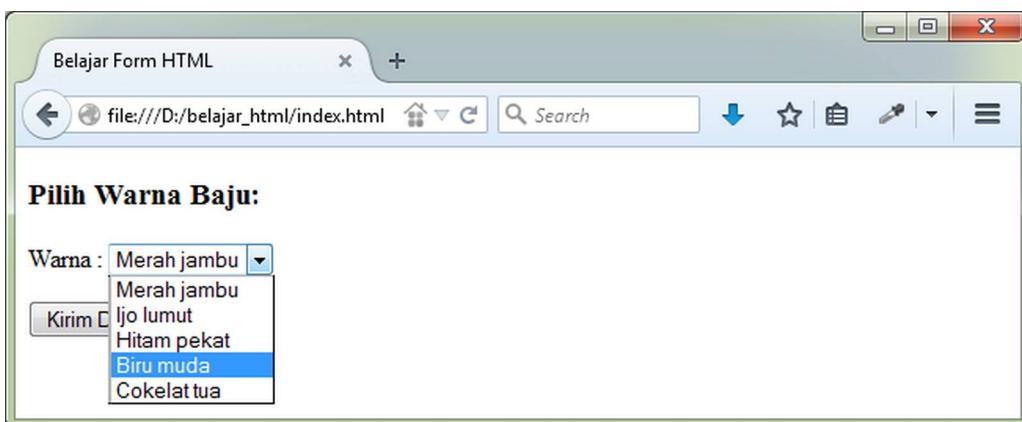
index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <h3>Pilih Warna Baju:</h3>
9 <form action="prosesform.php" method="get">
10 <p>Warna :
11   <select name="warna_baju">
12     <option value="merah">Merah jambu</option>
13     <option value="hijau">Ijo lumut</option>
14     <option value="hitam">Hitam pekat</option>
15     <option value="biru">Biru muda</option>
16     <option value="cokelat">Cokelat tua</option>
```

```

17  </select></p>
18  <p><input type="submit" value="Kirim Data"></p>
19 </form>
20 </body>
21 </html>

```



Gambar: Tampilan form `select`

Dalam kode di atas, tag `<select name="warna_baju">` adalah ‘*container*’ yang membungkus seluruh pilihan yang ada. Di dalam tag inilah atribut **name** ditempatkan.

Untuk membuat pilihan, kita menggunakan tag `<option>`. Teks pilihan yang tampil terletak diantara tag pembuka `<option>` dan tag penutup `</option>`, sedangkan yang dikirim ke server adalah nilai dari atribut `value`.

Sebagai contoh, jika saya memilih pilihan **Ijo lumut**, maka yang akan dikirim oleh web browser adalah:

```
file:///D:/belajar_html/prosesform.php?warna_baju=hijau
```

Pada alamat URL di atas, **warna_baju** berasal dari atribut **name** dari tag `<select>`, sedangkan nilai “**hijau**” berasal dari atribut `value` dari tag `<option>`.

Jika kita tidak membuat atribut `value`, maka nilai yang dikirim adalah teks diantara tag `<option>`, berikut contohnya:

index.html

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4    <meta charset="UTF-8">
5    <title>Belajar Form HTML</title>
6  </head>
7  <body>
8  <h3>Pilih Warna Baju:</h3>

```

```
9 <form action="prosesform.php" method="get">
10   <p>Warna :
11   <select name="warna_baju">
12     <option>Merah jambu</option>
13     <option>Ijo lumut</option>
14     <option>Hitam pekat</option>
15     <option>Biru muda</option>
16     <option>Cokelat tua</option>
17   </select></p>
18   <p><input type="submit" value="Kirim Data"></p>
19 </form>
20 </body>
21 </html>
```

Kali ini jika saya memilih warna “**Hitam pekat**” dan men-klik tombol submit, maka yang terkirim adalah **warna_baju=Hitam+pekat**, seperti yang terlihat di dalam URL berikut:

file:///D:/belajar_html/prosesform.php?warna_baju=Hitam+pekat

Atribut Selected

Kita bisa menambahkan atribut **selected** kepada tag `<option>` untuk membuat pilihan default ketika form ditampilkan. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Form HTML</title>
6   </head>
7   <body>
8     <h3>Pilih Warna Baju:</h3>
9     <form action="prosesform.php" method="get">
10       <p>Warna :
11         <select name="warna_baju">
12           <option value="merah">Merah jambu</option>
13           <option value="hijau">Ijo lumut</option>
14           <option value="hitam" selected>Hitam pekat</option>
15           <option value="biru">Biru muda</option>
16           <option value="cokelat">Cokelat tua</option>
17         </select>
18       </p>
19       <p><input type="submit" value="Kirim Data"></p>
```

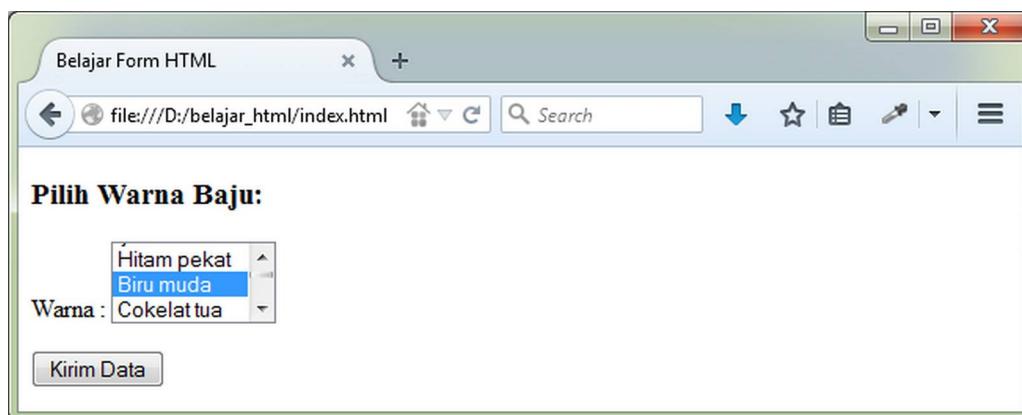
```
20 </form>
21 </body>
22 </html>
```

Kali ini, ketika form pertama kali ditampilkan warna “**Biru muda**” secara otomatis langsung terpilih.

Atribut **size** juga bisa ditambahkan ke dalam tag `<select>` untuk menentukan tinggi tampilan pilihan. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <h3>Pilih Warna Baju:</h3>
9 <form action="prosesform.php" method="get">
10 <p>Warna :
11   <select name="warna_baju" size="3">
12     <option value="merah">Merah jambu</option>
13     <option value="hijau">Ijo lumut</option>
14     <option value="hitam">Hitam pekat</option>
15     <option value="biru">Biru muda</option>
16     <option value="cokelat">Cokelat tua</option>
17   </select>
18 </p>
19 <p><input type="submit" value="Kirim Data"></p>
20 </form>
21 </body>
22 </html>
```



Gambar: Tampilan form `select` dengan atribut `size`

Jika anda menjalankan kode di atas, menu *dropdown* akan ditampilkan lebih ‘tinggi’ dari biasanya. Kali ini pilihan warna baju ditampilkan dengan tinggi 3 baris (sesuai dengan atribut `size="3"`). Jika atribut `size` tidak ditulis, nilai defaultnya adalah 1. Atribut `size` umumnya digunakan bersamaan dengan atribut **multiple**.

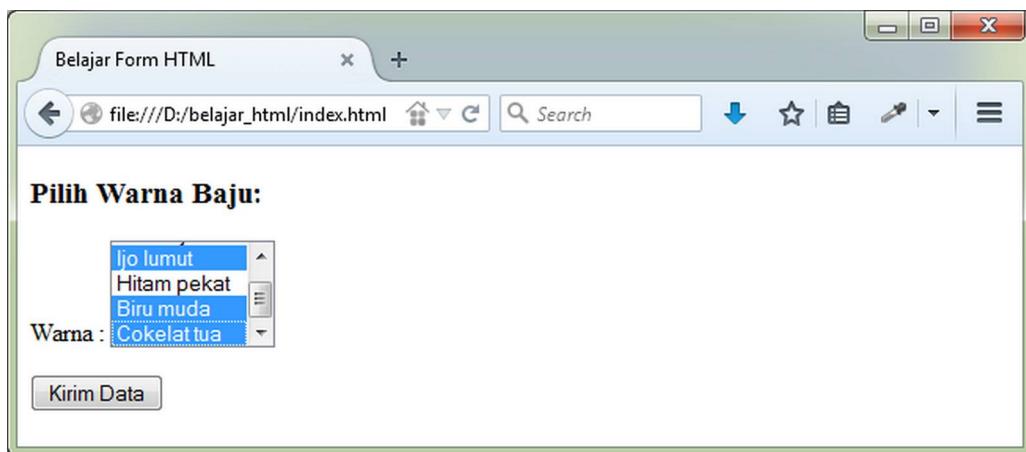
Atribut Multiple

Secara default, tag `<select>` hanya bisa dipilih 1 pilihan saja. Akan tetapi, jika kita menambahkan atribut **multiple**, maka pengguna bisa memilih lebih dari 1 pilihan. Untuk memilih lebih dari 1 pilihan, bisa menggunakan tombol **CTRL+klik**.

Berikut contoh penggunaan atribut multiple:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <h3>Pilih Warna Baju:</h3>
9 <form action="prosesform.php" method="get">
10  <p>Warna :
11    <select name="warna_baju" multiple>
12      <option value="merah">Merah jambu</option>
13      <option value="hijau">Ijo lumut</option>
14      <option value="hitam">Hitam pekat</option>
15      <option value="biru">Biru muda</option>
16      <option value="cokelat">Cokelat tua</option>
17    </select>
18  </p>
19  <p><input type="submit" value="Kirim Data"></p>
20 </form>
21 </body>
22 </html>
```

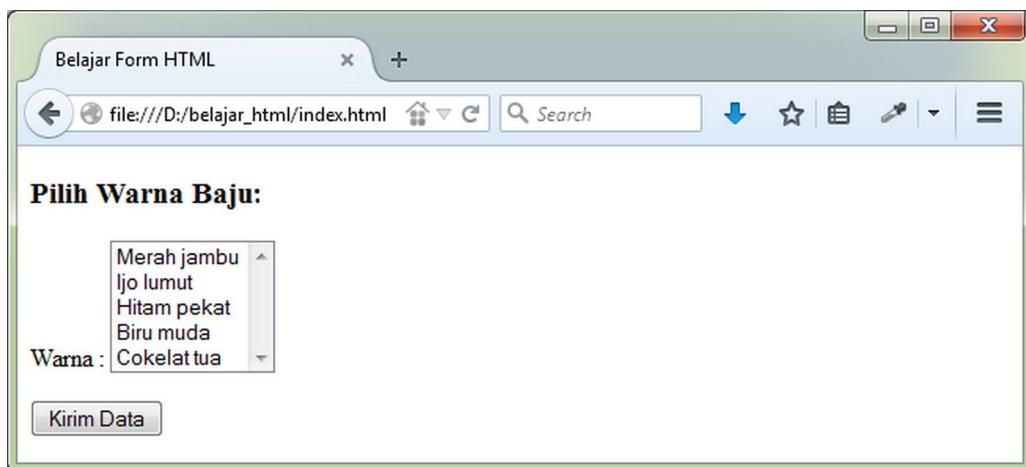


Gambar: Tampilan form `select` dengan atribut `multiple`

Seperti yang terlihat, web browser secara otomatis memperbesar baris pilihan (menjadi 4 baris) ketika kita menambahkan atribut `multiple`. Agar lebih rapi kita bisa mengatur manual tinggi dari tag `<select>` menggunakan atribut `size`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <h3>Pilih Warna Baju:</h3>
9 <form action="prosesform.php" method="get">
10 <p>Warna :
11   <select name="warna_baju" size="5" multiple>
12     <option value="merah">Merah jambu</option>
13     <option value="hijau">Ijo lumut</option>
14     <option value="hitam">Hitam pekat</option>
15     <option value="biru">Biru muda</option>
16     <option value="cokelat">Cokelat tua</option>
17   </select>
18 </p>
19 <p><input type="submit" value="Kirim Data"></p>
20 </form>
21 </body>
22 </html>
```



Gambar: Tampilan form `select` dengan atribut `multiple`

Kali ini saya menggunakan atribut `size="5"` agar seluruh pilihan dapat terlihat.

Walaupun sepertinya bermanfaat, menu dropdown dengan atribut multiple jarang digunakan. Karena tidak semua pengguna paham cara memilih menggunakan **CTRL + klik**. Objek form checkbox akan lebih sesuai untuk tujuan ini.

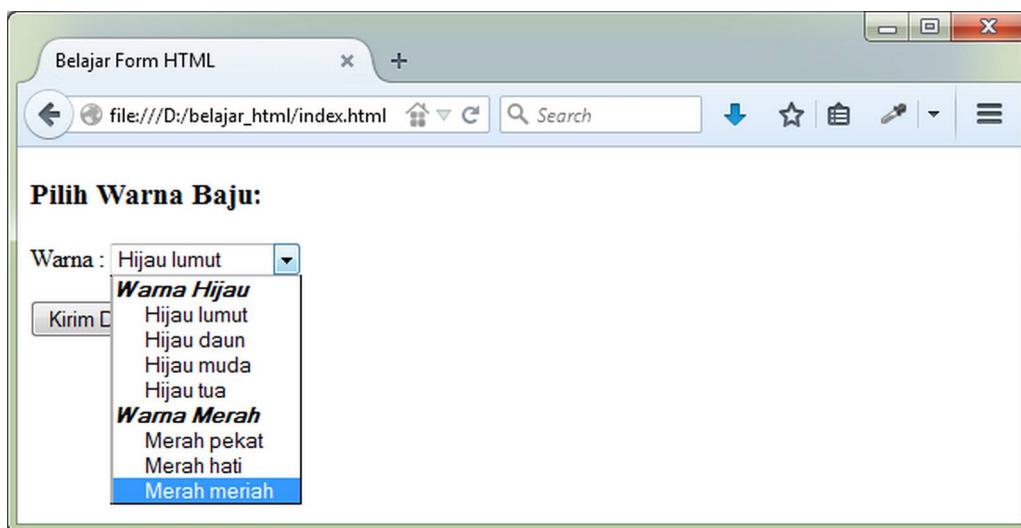
11.10 Optgroup Element

Untuk menyajikan data yang kompleks, kita bisa mengelompokkan pilihan tag `<option>` menggunakan tag `<optgroup>` dengan atribut `label`. Agar lebih mudah dipahami, berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <h3>Pilih Warna Baju:</h3>
9 <form action="prosesform.php" method="get" >
10   <p>Warna :
11     <select name="warna_baju" >
12       <optgroup label="Warna Hijau">
13         <option value="hijau_lumut">Hijau lumut</option>
14         <option value="hijau_daun">Hijau daun</option>
15         <option value="hijau_muda">Hijau muda</option>
16         <option value="hijau_tua">Hijau tua</option>
17     </optgroup>
18     <optgroup label="Warna Merah">
```

```
19      <option value="merah_pekat">Merah pekat</option>
20      <option value="merah_hati">Merah hati</option>
21      <option value="merah_meriah">Merah meriah</option>
22  </optgroup>
23  </select>
24  </p>
25  <p><input type="submit" value="Kirim Data"></p>
26 </form>
27 </body>
28 </html>
```



Gambar: Tampilan form `select` dengan tag `optgroup`

Dalam contoh di atas, saya membuat 2 kelompok warna, "Warna Hijau" dan "Warna Merah". Pengelompokan ini dibuat menggunakan tag `<optgroup>`. Atribut `label` berfungsi untuk menampilkan judul pilihan.

Di dalam web browser, tag `<optgroup>` ditampilkan terpisah dan tidak bisa dipilih (karena memang bukan bagian dari pilihan). Tampilan seperti ini cocok digunakan jika data cukup panjang, sehingga akan memudahkan bagi pengguna.

11.11 Input Element Type File

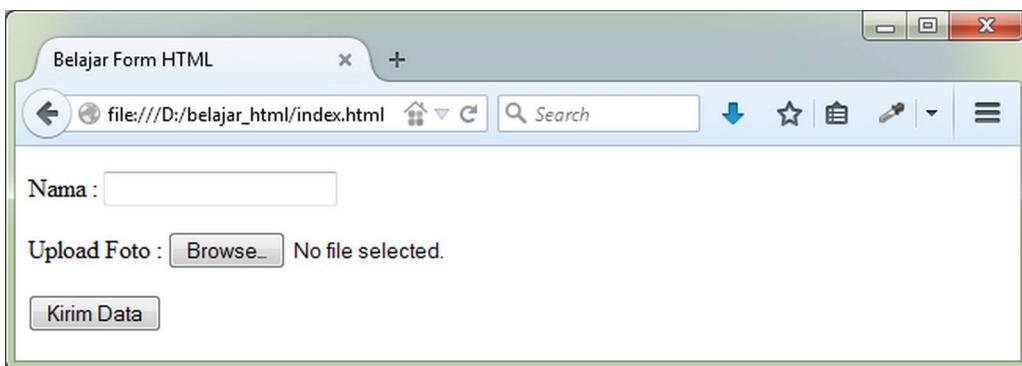
Untuk mengupload sebuah file, kita menggunakan tag `<input>` dengan atribut `type="file"`. Tag ini digunakan untuk men-upload sesuatu ke server, apakah itu berupa gambar, musik, video, dokumen, dll.

Berikut contoh penggunaannya:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="post">
9   <p>Nama : <input type="text" name="nama"></p>
10  <p>Upload Foto : <input type="file" name="file_gambar"></p>
11  <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>
```

Gambar: Tampilan form `input type="file"`

Atribut Enctype

Agar dapat diproses oleh server, tag `<form>` harus dikirim menggunakan `method="post"` dan tambahan atribut `enctype="multipart/form-data"`. Ini berkaitan dengan cara pengiriman data ke server.

Sebenarnya atribut `enctype` tetap digunakan oleh web browser. Jika kita tidak menulis atribut ini, web browser menambahkan atribut `enctype="application/x-www-form-urlencoded"`. Perbedaan antara keduanya terdapat pada proses ‘encoding’ ketika form dikirim.

Jika kita menggunakan `enctype="application/x-www-form-urlencoded"` (pilihan default), karakter-khusus akan dikonversi (dalam istilah pemrograman: di-**encode**) menjadi angka karakter ASCII, dan tanda spasi akan diganti menjadi tanda ‘+’.

Apabila menggunakan `enctype="multipart/form-data"`, form akan dikirim “apa adanya” tanpa dilakukan proses encode. Untuk form yang memiliki fitur upload, kita harus menggunakan `enctype="multipart/form-data"` ini. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="post" enctype="multipart/form-data">
9   <p>Nama : <input type="text" name="nama"></p>
10  <p>Upload Foto : <input type="file" name="file_gambar"></p>
11  <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>
```

Proses penanganan file upload selanjutnya dilakukan di sisi server.

Atribut Accept

Mirip dengan atribut **maxlength** yang membatasi jumlah karakter inputan, untuk objek form upload, kita bisa menggunakan atribut **accept**. Atribut accept akan membatasi jenis-jenis file yang akan diupload, apakah itu file gambar saja, file dokumen, atau pembatasan lainnya.

Nilai untuk atribut accept berupa **MIME_type** seperti **audio/***, **video/*** atau **image/***. Berikut contoh penulisannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="post" enctype="multipart/form-data">
9   <p>Nama : <input type="text" name="nama"></p>
10  <p>Upload Foto : <input type="file" name="file_gambar" accept="image/*"></p>
11  <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>
```

Form di atas hanya akan menerima upload berupa file gambar, sesuai dengan atribut `accept="image/*"`.

Akan tetapi pembatasan dari HTML seperti ini sebaiknya kembali diperiksa di sisi server, karena bisa dilewati dengan mudah.

11.12 Input Element Type Hidden

HTML juga menyediakan objek form yang tidak terlihat (*hidden*). Untuk membuatnya, kita menggunakan atribut `type="hidden"` pada tag `<input>`. Seluruh atribut lain seperti `name`, `value`, `disabled`, dll juga berfungsi di dalam objek form ini.

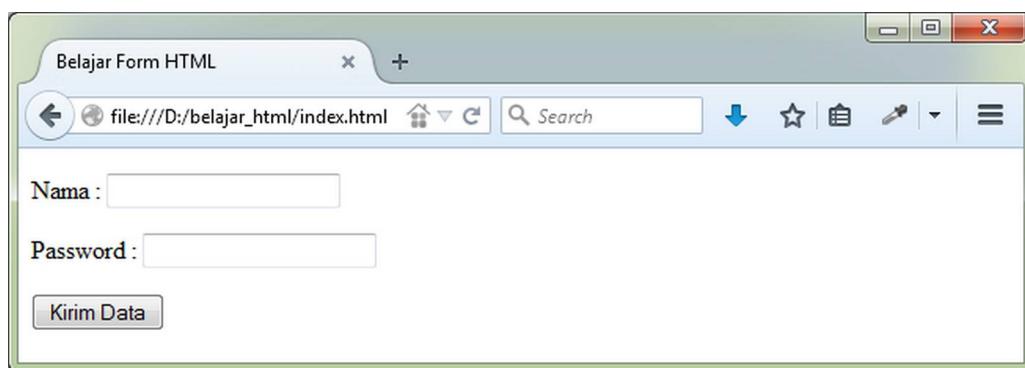
Berikut contoh penggunaan input element type hidden:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="nama"></p>
10  <p>Password : <input type="password" name="password" /></p>
11  <input type="hidden" name="tersembunyi"/>
12  <p><input type="submit" value="Kirim Data"></p>
13 </form>
14 </body>
15 </html>

```



Gambar: Tampilan form `input type="hidden"`

Anda mungkin bertanya, jika objek form ini tidak terlihat, jadi apa fungsinya?

Tag `<input type="hidden">` bisa digunakan untuk mengirim ‘pesan rahasia’ yang digunakan untuk proses konfirmasi form. Sebagai contoh, perhatikan kode berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="nama"></p>
10  <p>Password : <input type="password" name="password" /></p>
11  <input type="hidden" name="token" value="h68et56d6"/>
12  <p><input type="submit" value="Kirim Data"></p>
13 </form>
14 </body>
15 </html>
```

Dalam form di atas saya menambahkan sebuah objek form hidden dengan `name="token"` dan `value="h68et56d6"`. Kode token ini bisa dibuat secara otomatis oleh server, dan digunakan untuk konfirmasi bahwa form memang berasal dari website kita (mencegah form diinput dari lokasi lain).

11.13 Input Element Type Image

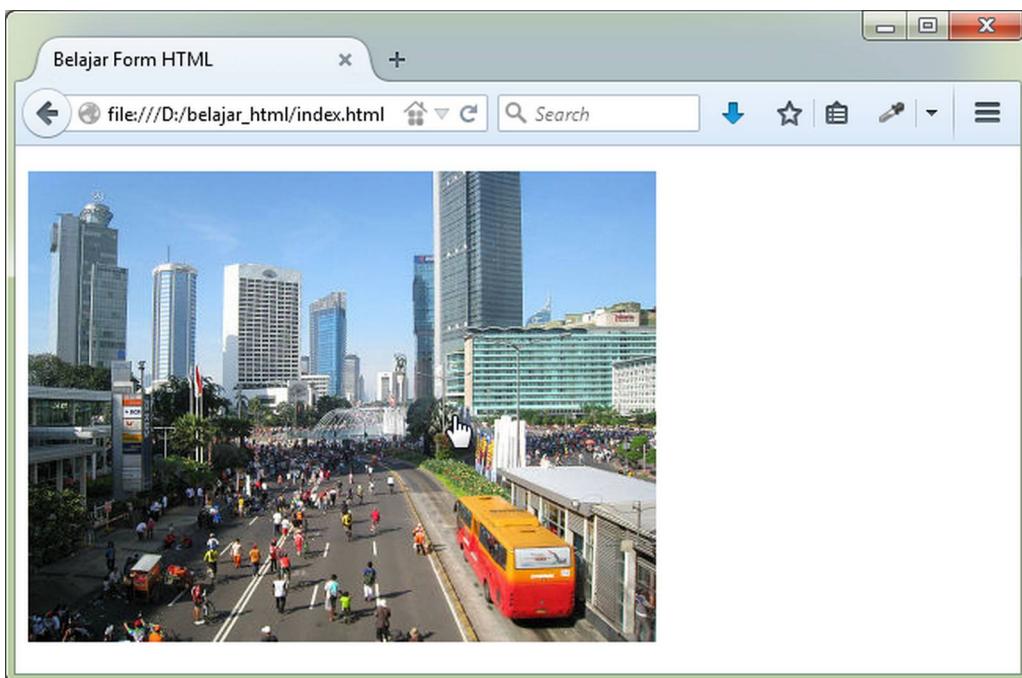
Kita juga bisa menggunakan gambar sebagai objek input form. Ini bisa digunakan untuk mengganti tombol submit standar bawaan web browser, atau membuat aplikasi yang perhubungan dengan titik koordinat, seperti peta atau grafik matematis.

Untuk keperluan ini, HTML menyediakan atribut `type="image"` untuk tag `<input>`. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>
10    <input type="image" name="gambar" src="Bundaran_HI.jpg">
11   </p>
12 </form>
```

```
13 </body>
14 </html>
```



Gambar: Tampilan form `input type="image"`

Perhatikan bahwa pada tag `input type image`, juga harus menyertakan atribut `src` yang berisi alamat gambar. Jika sebuah area gambar di klik, posisi cursor akan dikirim ke server. Sebagai contoh, ketika saya men-klik gambar roda belakang busway, web browser akan mengirim data berikut ini:

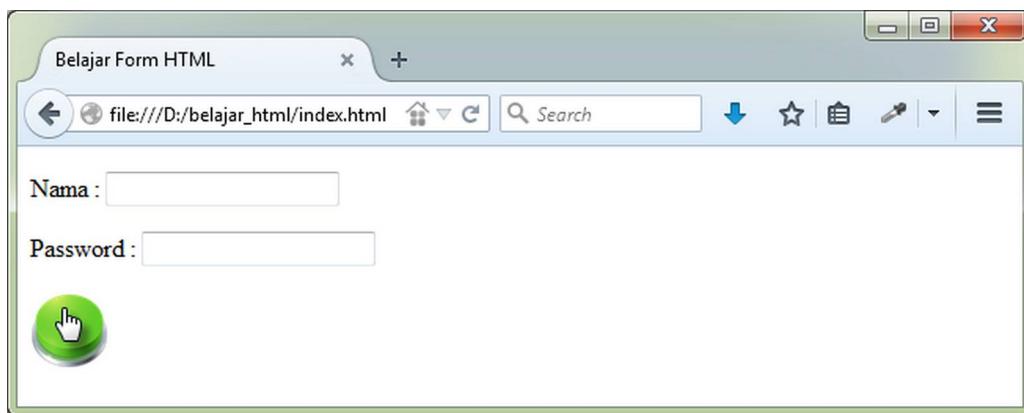
```
file:///D:/belajar_html/prosesform.php?gambar.x=291&gambar.y=268
```

gambar.x dan **gambar.y** adalah titik koordinat gambar. Perhitungan koordinat dimulai dari (0,0) di sisi kiri atas, hingga titik (400,300) di sisi kanan bawah, ini karena ukuran gambar yang saya gunakan adalah 400×300 pixel.

Selain untuk mencatat koordinat gambar, tag `input type image` juga bisa digunakan untuk mengganti tombol submit form, seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="nama"></p>
10  <p>Password : <input type="password" name="password" ></p>
11  <p><input type="image" name="kirim" src="tombol.png" width="50"></p>
12 </form>
13 </body>
14 </html>
```



Gambar: Tampilan form `input type="image"`

Pada contoh di atas, tombol submit bisa digantikan dengan gambar menggunakan tag `input type="image"`.

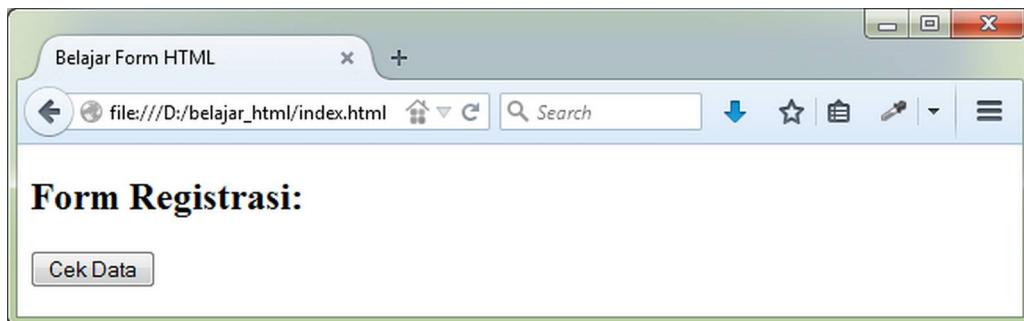
11.14 Input Element Type Button

Selain menggunakan tombol submit dan reset, kita juga bisa menambahkan tombol yang didefinisikan sendiri. Untuk keperluan ini HTML menyediakan atribut `type="button"` pada tag `<input>`.

Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <h2>Form Registrasi:</h2>
9 <form action="prosesform.php" method="get">
10  <p><input type="button" value="Cek Data"></p>
11 </form>
12 </body>
13 </html>
```

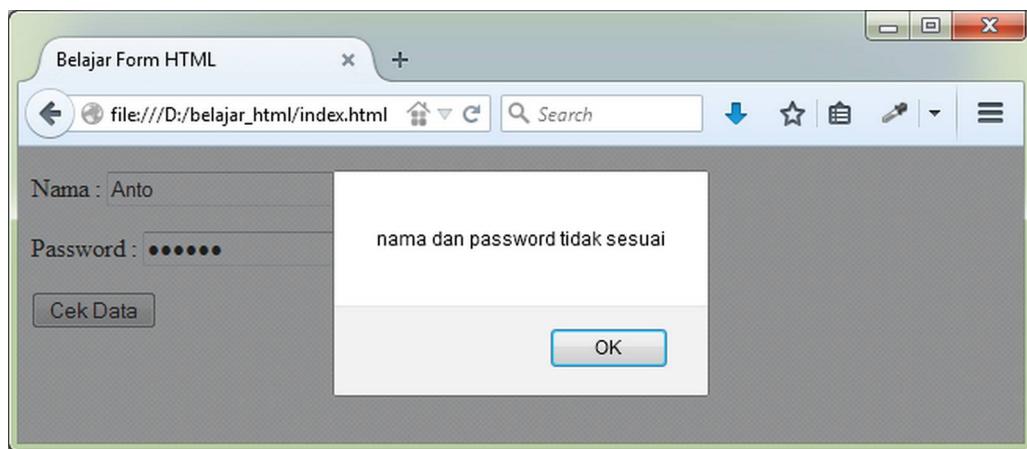


Gambar: Tampilan form `input type="button"`

Namun berbeda dengan tombol submit dan reset, tombol ini tidak memiliki fungsi bawaan. Kita bisa mendefinisikan sendiri fungsi tombol ini menggunakan JavaScript, seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="nama"></p>
10  <p>Password : <input type="password" name="password" /></p>
11  <p><input type="button" value="Cek Data"
12    onClick="alert('nama dan password tidak sesuai')"/></p>
13 </form>
14 </body>
15 </html>
```



Gambar: Tampilan form `input type="button"` dengan tambahan JavaScript

Kali ini tombol Cek Data akan menampilkan pesan ‘*nama dan password tidak sesuai*’ dengan menggunakan JavaScript.

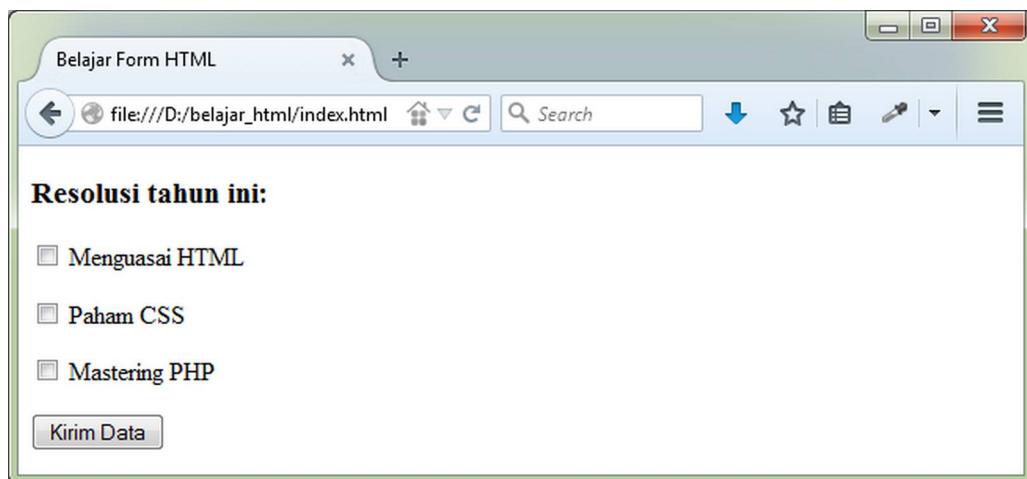
11.15 Label Element

Salah satu tag HTML yang sering digunakan di dalam form adalah tag `<label>`. Tag ini tidak mempengaruhi tampilan, namun akan membuat interaksi dengan form menjadi lebih baik (*user friendly*).

Sebagai contoh, perhatikan form berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Form HTML</title>
6   </head>
7   <body>
8     <h3>Resolusi tahun ini:</h3>
9     <form action="prosesform.php" method="get">
10       <p><input type="checkbox" name="target1" value="HTML"> Menguasai HTML<p>
11       <p><input type="checkbox" name="target2" value="CSS"> Paham CSS<p>
12       <p><input type="checkbox" name="target3" value="PHP"> Mastering PHP<p>
13       <p><input type="submit" value="Kirim Data"><p>
14     </form>
15   </body>
16 </html>
```

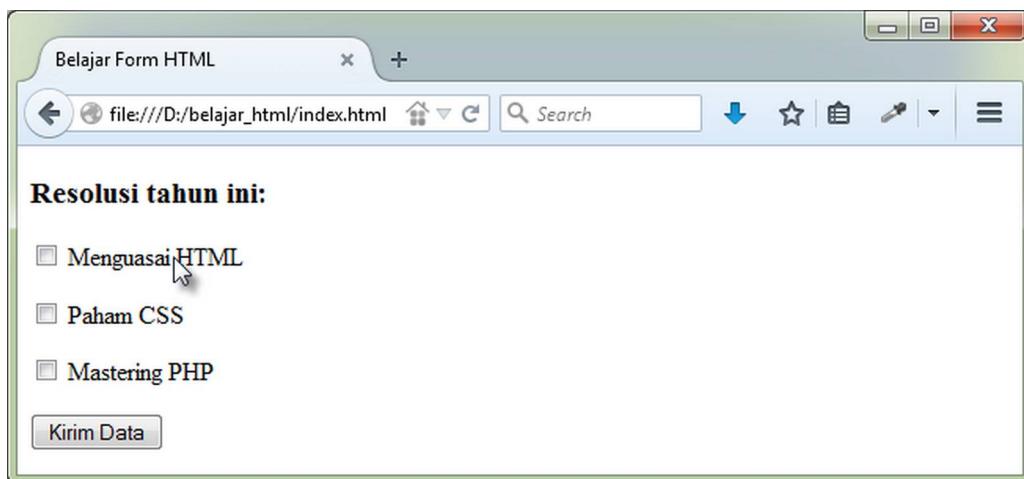


Gambar: Tampilan form dengan checkbox

Untuk men-klik tombol checkbox, kita harus menempatkan mouse persis di atas kotak checkbox. Akan lebih nyaman jika pengguna bisa memilih dengan men-klik tulisan keterangan. Untuk hal inilah kita menggunakan tag <label>:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <h3>Resolusi tahun ini:</h3>
9 <form action="prosesform.php" method="get">
10  <p><label><input type="checkbox" name="target1" value="HTML">
11    Menguasai HTML</label></p>
12  <p><label><input type="checkbox" name="target2" value="CSS">
13    Paham CSS</label></p>
14  <p><label><input type="checkbox" name="target3" value="PHP">
15    Mastering PHP</label></p>
16  <p><label><input type="submit" value="Kirim Data"></label></p>
17 </form>
18 </body>
19 </html>
```



Gambar: Tampilan form dengan tag `label`

Kali ini, kita bisa men-klik tulisan keterangan dan secara otomatis checkbox akan langsung terpilih (tanpa harus men-klik tepat di atas kotak check box).

Selain ‘mengurung’ objek form dengan tag `<label>`, kita juga bisa memisahkan penulisannya dengan bantuan atribut `for`. Sehingga kode di atas dapat juga ditulis seperti berikut ini:

index.html

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Form HTML</title>
6   </head>
7   <body>
8     <h3>Resolusi tahun ini:</h3>
9     <form action="prosesform.php" method="get">
10       <p><input type="checkbox" name="target1" value="HTML" id="html">
11         <label for="html"> Menguasai HTML</label></p>
12       <p><input type="checkbox" name="target2" value="CSS" id="css">
13         <label for="css"> Paham CSS</label></p>
14       <p><input type="checkbox" name="target3" value="PHP" id="php">
15         <label for="php"> Mastering PHP</label></p>
16       <p><input type="submit" value="Kirim Data"></p>
17     </form>
18   </body>
19 </html>

```

Kali ini tag `<label>` hanya berisi teks keterangan dari checkbox. Dengan menggunakan atribut `for`, kita bisa menghubungkan label dengan checkbox yang sesuai.

Agar keduanya bisa terhubung, nilai dari atribut `for` harus cocok dengan atribut `id` pada objek form. Sebagai contoh, jika checkbox memiliki atribut `id="php"`, maka pada tag label harus ditulis `for="php"`.

Penggunaan tag `<label>` tidak hanya terbatas untuk checkbox saja, kita juga bisa menggunakan-nya untuk objek form lain seperti *radio button*, *text box*, dll. Selain itu, penambahan tag ini juga bisa digunakan sebagai selector CSS.

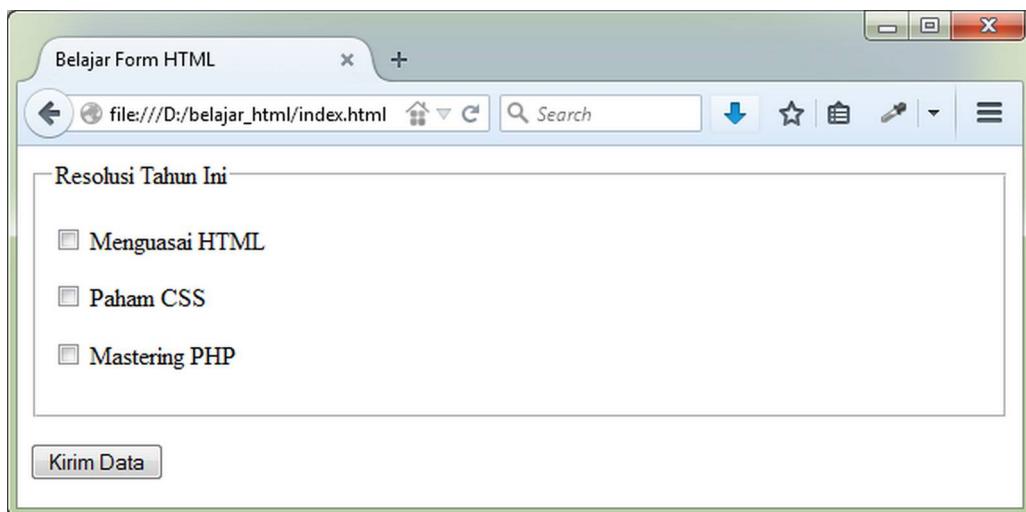
11.16 Fieldset Element dan Legend Element

Tag `<fieldset>` dan tag `<legend>` digunakan untuk ‘mempercantik’ tampilan form dengan mengelompokkan objek form sesuai dengan pembagiannya. Tag `<fieldset>` berfungsi sebagai ‘container’ yang menampung objek form, sedangkan tag `<legend>` berfungsi untuk membuat judul penjelasan.

Berikut contoh penggunaan tag `<fieldset>` dan tag `<legend>`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9 <fieldset>
10 <legend>Resolusi Tahun Ini</legend>
11   <p><input type="checkbox" name="target1" value="HTML"> Menguasai HTML</p>
12   <p><input type="checkbox" name="target2" value="CSS"> Paham CSS</p>
13   <p><input type="checkbox" name="target3" value="PHP"> Mastering PHP</p>
14 </fieldset>
15   <p><input type="submit" value="Kirim Data"></p>
16 </form>
17 </body>
18 </html>
```

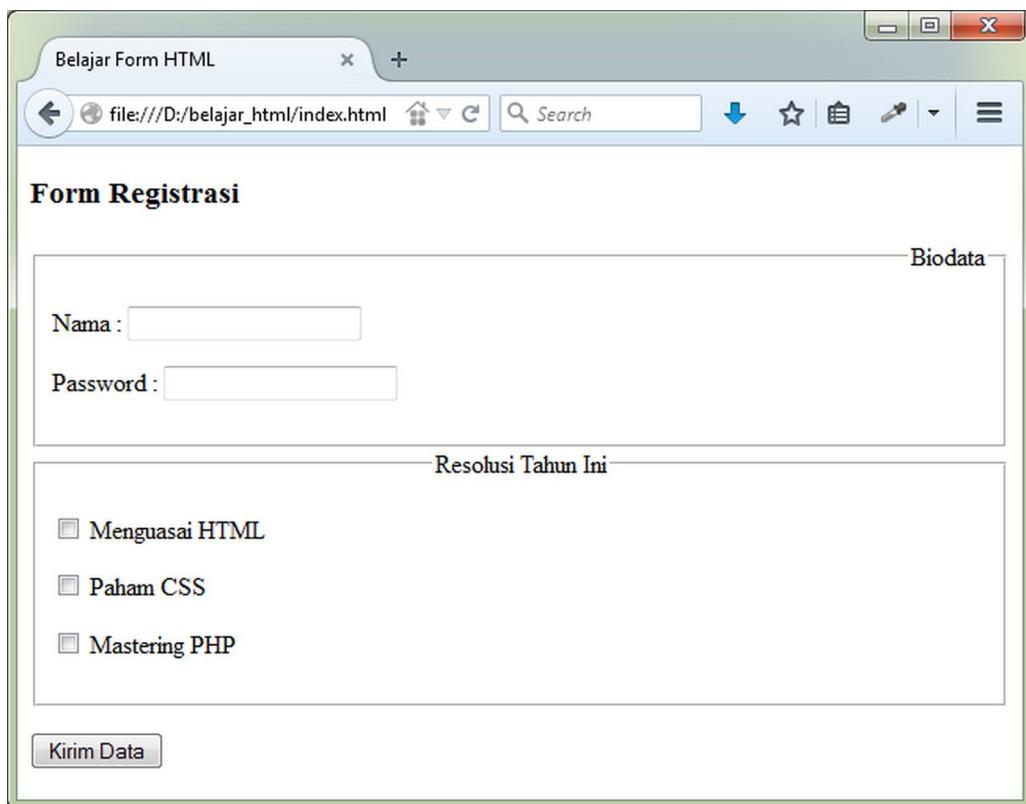


Gambar: Tampilan form dengan tag legend

Tag <legend> juga memiliki atribut align yang bisa digunakan untuk mengatur posisi teks. Nilai yang bisa gunakan adalah *left*, *center*, *right*, *top* dan *bottom*. Namun nilai top dan bottom umumnya akan diabaikan oleh web browser.

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Form HTML</title>
6   </head>
7   <body>
8     <h3>Form Registrasi</h3>
9     <form action="prosesform.php" method="get">
10    <fieldset>
11      <legend align="right">Biodata</legend>
12      <p>Nama : <input type="text" name="nama"></p>
13      <p>Password : <input type="password" name="password"></p>
14    </fieldset>
15    <fieldset>
16      <legend align="center">Resolusi Tahun Ini</legend>
17      <p><input type="checkbox" name="target1" value="HTML"> Menguasai HTML</p>
18      <p><input type="checkbox" name="target2" value="CSS"> Paham CSS</p>
19      <p><input type="checkbox" name="target3" value="PHP"> Mastering PHP</p>
20    </fieldset>
21    <p><input type="submit" value="Kirim Data"></p>
22  </form>
23 </body>
24 </html>
```



Gambar: Tampilan form dengan tag **legend** dan atribut **align**

11.17 Button Element

Selain menggunakan tag `<input type="submit">` dan `<input type="button">`, kita juga bisa membuat tombol dengan tag `<button>`.

Berbeda dengan tag `<input type="submit">`, tombol dengan `<button>` element memerlukan tag penutup `</button>`, dan yang akan ditampilkan sebagai teks di dalam tombol adalah teks diantaranya bukan isi dari atribut `value`.

Sebagai contoh, untuk membuat tombol dengan nilai "Kirim Data", ditulis menjadi:

```
<button>Kirim Data</button>
```

Apabila tag `<button>` diletakkan di dalam form, secara default fungsinya juga untuk men-submit form, walaupun di dalam form sudah ada tag `<input type="submit">`.

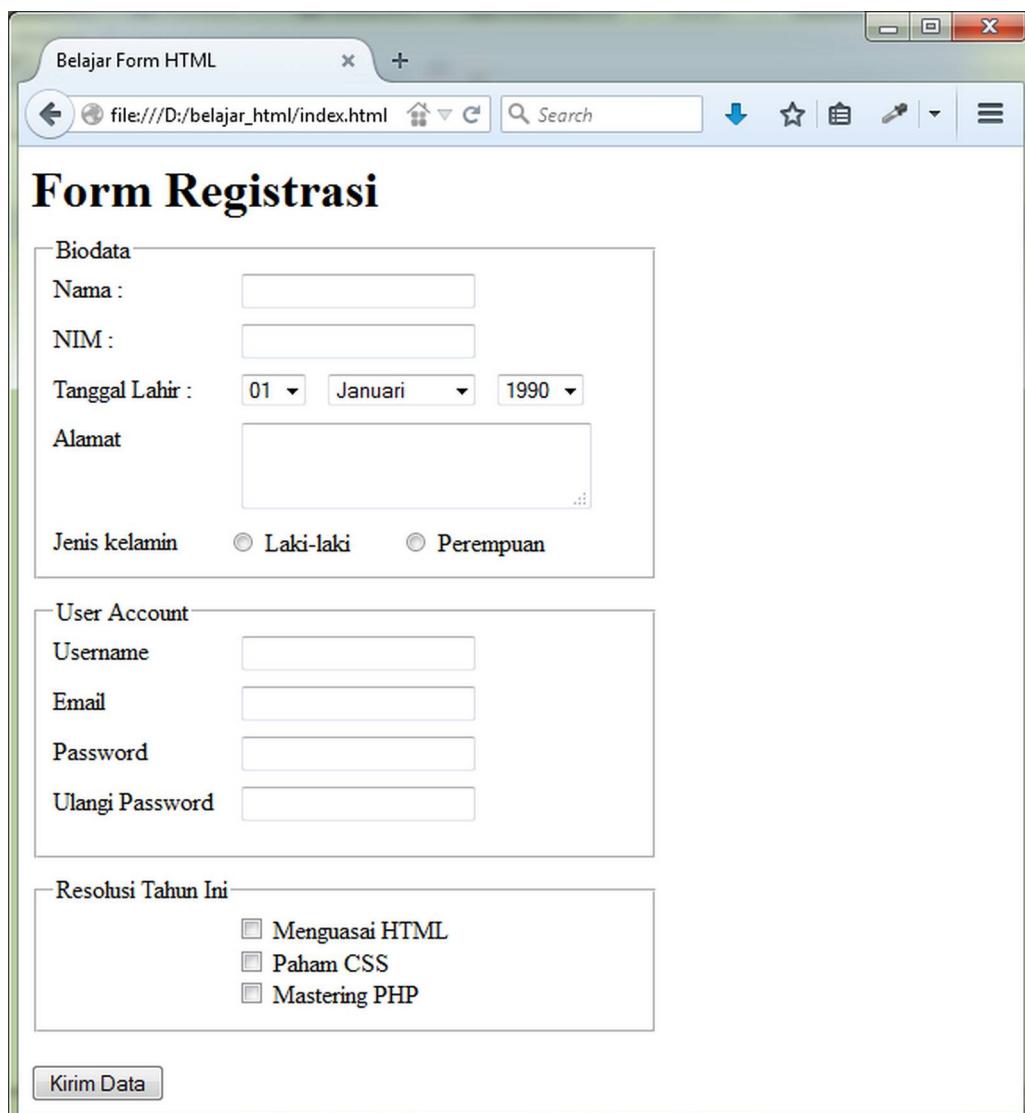
Umumnya tag `<button>` digunakan untuk membuat tombol yang berada di luar form. Tombol yang terletak bukan di dalam form tidak memiliki fungsi bawaan, sehingga cocok digunakan sebagai 'pemanggil' kode JavaScript.

11.18 Penutup: Form Element

Dalam bab ini kita telah membahas banyak objek form yang disediakan dalam HTML. Sebagai penutup, berikut adalah sebuah form utuh yang menyertakan sebagian besar dari apa yang telah

kita pelajari.

Berikut adalah tampilan form:



Gambar: Tampilan form registrasi, dengan berbagai objek form

Dan berikut adalah kode program yang diperlukan:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML </title>
6   <style>
7     h1 {
8       margin:0;
9     }
```

```
10     div {
11         width:400px;
12     }
13     p{
14         margin:0;
15     }
16     fieldset{
17         margin-top:10px;
18     }
19     input[type="text"], input[type="password"], select, textarea {
20         margin-left:10px;
21         margin-bottom:10px;
22     }
23     input[type="checkbox"] {
24         margin-left:120px;
25     }
26     label {
27         width:110px;
28         float:left;
29     }
30     label[for="html"], label[for="css"],label[for="php"] {
31         float: initial;
32     }
33 </style>
34 </head>
35 <body>
36
37 <div>
38 <h1>Form Registrasi</h1>
39
40 <form action="prosesform.php" method="get">
41 <fieldset>
42 <legend>Biodata</legend>
43     <p>
44         <label for="nama">Nama : </label>
45         <input type="text" name="nama" id="nama">
46     </p>
47     <p>
48         <label for="nim">NIM : </label>
49         <input type="text" name="nim" id="nim">
50     </p>
51     <p>
52         <label for="tgl" >Tanggal Lahir : </label>
53         <select name="tgl" id="tgl">
54             <option value=1>01</option>
55             <option value=2>02</option>
```

```
56      <option value=3>03</option>
57      <option value=4>04</option>
58      <option value=5>05</option>
59      <option value=6>06</option>
60      <option value=7>07</option>
61      <option value=8>08</option>
62      <option value=9>09</option>
63      <option value=10>10</option>
64      <option value=11>11</option>
65      <option value=12>12</option>
66      <option value=13>13</option>
67      <option value=14>14</option>
68      <option value=15>15</option>
69      <option value=16>16</option>
70      <option value=17>17</option>
71      <option value=18>18</option>
72      <option value=19>19</option>
73      <option value=20>20</option>
74      <option value=21>21</option>
75      <option value=22>22</option>
76      <option value=23>23</option>
77      <option value=24>24</option>
78      <option value=25>25</option>
79      <option value=26>26</option>
80      <option value=27>27</option>
81      <option value=27>28</option>
82      <option value=29>29</option>
83      <option value=30>30</option>
84      <option value=31>31</option>
85  </select>
86  <select name="bln">
87      <option value=1>Januari</option>
88      <option value=2>Februari</option>
89      <option value=3>Maret</option>
90      <option value=4>April</option>
91      <option value=5>Mei</option>
92      <option value=6>Juni</option>
93      <option value=7>Juli</option>
94      <option value=8>Agustus</option>
95      <option value=9>September</option>
96      <option value=10>Oktober</option>
97      <option value=11>Nopember</option>
98      <option value=12>Desember</option>
99  </select>
100 <select name="thn">
101    <option value=1>1990</option>
```

```
102      <option value=2>1991</option>
103      <option value=3>1992</option>
104      <option value=4>1993</option>
105      <option value=5>1994</option>
106      <option value=6>1995</option>
107      <option value=7>1996</option>
108      <option value=8>1997</option>
109      <option value=9>1998</option>
110      <option value=10>1999</option>
111      <option value=11>2000</option>
112    </select>
113  </p>
114  <p>
115    <label for="alamat">Alamat </label>
116    <textarea name="alamat" id="alamat" cols="25"></textarea>
117  </p>
118  <p>
119    <label>Jenis kelamin</label>
120    <label><input type="radio" name="kel" value="laki2"> Laki-laki</label>
121    <label><input type="radio" name="kel" value="perempuan"> Perempuan</label>
122  </p>
123 </fieldset>
124
125 <fieldset>
126 <legend align="">User Account</legend>
127  <p>
128    <label for="username">Username</label>
129    <input type="text" name="username" id="username"/>
130  </p>
131  <p>
132    <label for="email">Email </label>
133    <input type="text" name="email" id="email" />
134  </p>
135  <p>
136    <label for="pass">Password</label>
137    <input type="password" name="password" id="pass" />
138  </p>
139  <p>
140    <label for="repass">Ulangi Password</label>
141    <input type="password" name="repassword" id="repass" />
142  </p>
143 </fieldset>
144
145 <fieldset>
146 <legend>Resolusi Tahun Ini</legend>
147  <p>
```

```
148     <input type="checkbox" name="target1" value="HTML" id="html">
149     <label for="html"> Menguasai HTML</label>
150 </p>
151 <p>
152     <input type="checkbox" name="target2" value="CSS" id="css">
153     <label for="css"> Paham CSS</label>
154 </p>
155 <p>
156     <input type="checkbox" name="target3" value="PHP" id="php">
157     <label for="php"> Mastering PHP</label>
158 </p>
159 </fieldset>
160 <br>
161 <p>
162     <input type="submit" value="Kirim Data">
163 </p>
164 </form>
165
166 </div>
167
168 </body>
169 </html>
```

Anda dapat abaikan bagian CSS dari kode di atas, ini digunakan hanya untuk merapikan tampilan form. Silahkan pelajari bagaimana cara penulisan setiap objek form. Jika anda sudah memahami bab ini, tentunya tidak kesulitan untuk mengerti fungsi dari setiap tag dan atribut yang digunakan.

Jika kita mengisi form tersebut dan men-klik tombol “Kirim Data”, semua nilai form akan dikirim ke server. Sebagai contoh, berikut adalah hasil URL pengisian form tersebut:

```
file: //D:/belajar_html/prosesform.php?nama=Andika+Sofyana&nim=1014070030
&tgl=4&bln=5&thn=1&alamat=Jl.+Perintis+Kemerdekaan+No.1+Medan
&kel=perempuan&username=andika10&email=dika%40duniailkom.com
&password=qwertyuu&repassword=qwertyu&target1=HTML&target2=CSS
```

Setiap nilai form ini dapat diproses dengan menggunakan bahasa pemrograman server seperti PHP, untuk kemudian disimpan di dalam database, seperti MySQL Server.

Dalam bab kali ini kita telah mempelajari sebagian besar tag dan atribut yang berhubungan dengan pembuatan form di dalam HTML.

Walaupun begitu, banyak pihak yang masih merasa kurang dengan fitur-fitur tersebut. HTML5 hadir dengan berbagai fitur terbaru untuk membuat form, seperti objek form baru dan proses validasi. Dalam tutorial berikutnya kita akan membahas tentang hal ini dengan lebih dalam.

12. HTML5 Form Element

Setelah membahas tentang cara pembuatan form HTML dasar dalam bab sebelumnya, kali ini kita akan fokus kepada fitur-fitur form terbaru dari HTML5. HTML5 membawa berbagai objek form dan atribut baru, serta fitur validasi tanpa menggunakan JavaScript.

12.1 Validasi Form

Validasi menjadi aspek penting dalam setiap form. Validasi adalah proses untuk memastikan data inputan form telah sesuai dengan yang diharapkan. Misalnya, untuk form umur seharusnya hanya diisi dengan angka, akan tetapi pengguna yang ‘usil’ atau sedang terburu-buru bisa saja men-input huruf ke dalam inputan ini.

Proses validasi biasanya dilakukan dua kali, yakni di sisi client (web browser) menggunakan JavaScript, dan di sisi server (menggunakan PHP).

Validasi di sisi server relatif lebih aman, karena pengguna tidak bisa men-interupsi proses yang terjadi. Akan tetapi agar dapat diproses, data form harus dikirim terlebih dahulu ke server, dan jika ditemukan kesalahan, form kembali ditampilkan ke web browser. Alur seperti ini terus dilakukan hingga seluruh data form lolos validasi. Jika koneksi internet lambat, proses ini bisa memakan waktu yang tidak sebentar.

Dilain pihak, validasi menggunakan JavaScript berjalan di web browser, sehingga dieksekusi pada saat itu juga. Namun proses ini bisa diinterupsi oleh pengguna (misalnya dengan mematikan JavaScript). Oleh karena itu validasi di sisi server tetap diperlukan.

Membuat validasi membutuhkan waktu yang tidak sebentar, namun sangat penting. Oleh karena itulah WHATWG dan W3C menghadirkan fitur validasi form ke dalam HTML5. Walaupun tidak sepenuhnya menggantikan peran JavaScript, fitur ini sangat menarik.

Perlu untuk diingat tidak semua web browser mendukung HTML5. Bagi web browser yang tidak support, proses validasi tidak akan berjalan.

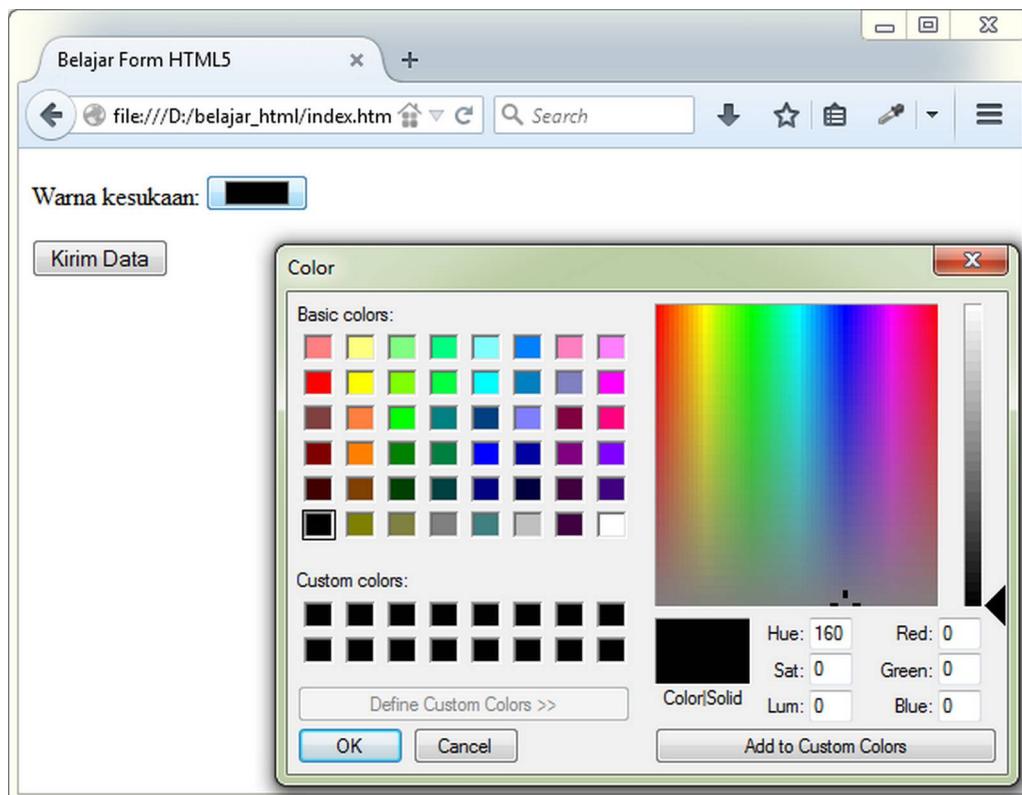
12.2 Input Element Type Color

HTML5 menyediakan objek form khusus untuk memilih warna, yakni menggunakan atribut `type="color"` pada tag `<input>`. Ketika di pilih, objek form ini akan menampilkan ‘jendela warna’ (*color picker*) tergantung dengan jenis web browser dan sistem operasi yang digunakan.

Sebagai contoh, berikut adalah cara penggunaan `input type color`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Warna kesukaan: <input type="color" name="warna"></p>
10  <p><input type="submit" value="Kirim Data"></p>
11 </form>
12 </body>
13 </html>
```



Gambar: Tampilan form `input type="color"`

Ketika objek form dipilih, nilai yang dikirim adalah angka *heksadesimal* warna dengan format `#RRGGBB`. Jika anda sudah pernah mempelajari CSS, tentunya tidak asing dengan format warna ini. R mewakili warna merah (Red), G warna hijau (Green) dan B adalah warna biru (Blue).

Sebagai contoh, ketika saya memilih warna merah, berikut adalah hasil yang didapat:

```
file:///D:/belajar_html/prosesform.php?warna=%23ff0000
```

Karakter %23 adalah hasil konversi tanda '#' menjadi karakter ASCII.

12.3 Input Element Type Email

Email adalah salah satu isian form yang hampir selalu diperlukan dalam setiap form registrasi / login. Oleh karena hal inilah HTML5 menyediakan atribut type="email" untuk tag <input>.

Objek form **input type email** tampak seperti text box reguler, akan tetapi web browser menambahkan proses validasi untuk memeriksa isian form agar memenuhi format alamat email.

Sebuah alamat email yang valid, dimulai dengan beberapa karakter, kemudian diikuti oleh tanda '@' dan diakhiri dengan nama domain, seperti yahoo.com, atau gmail.com. Jika pengguna tidak menginput sesuai dengan format ini, form akan menampilkan pesan error ketika form di-submit.

Berikut contoh penulisan tag **input type email**:

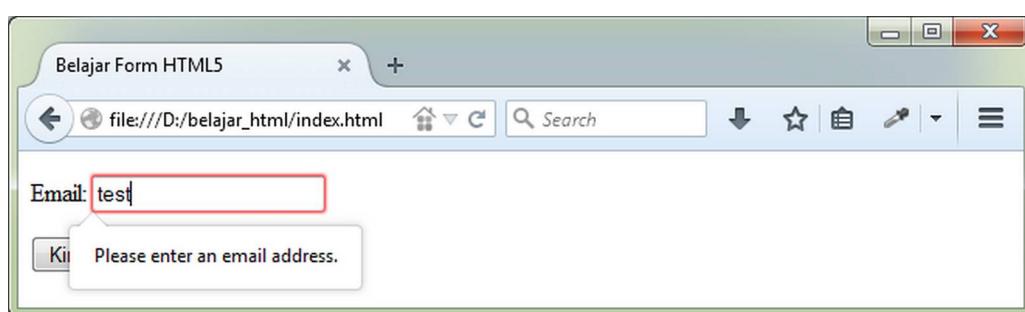
index.html

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Belajar Form HTML5</title>
6  </head>
7  <body>
8      <form action="prosesform.php" method="get">
9          <p>Email: <input type="email" name="email"></p>
10         <p><input type="submit" value="Kirim Data"></p>
11     </form>
12 </body>
13 </html>
```

Untuk web browser yang mendukung HTML5, proses validasi akan menampilkan pesan error.

Berikut contohnya di dalam Mozilla Firefox:



Gambar: Tampilan form **input type="email"**

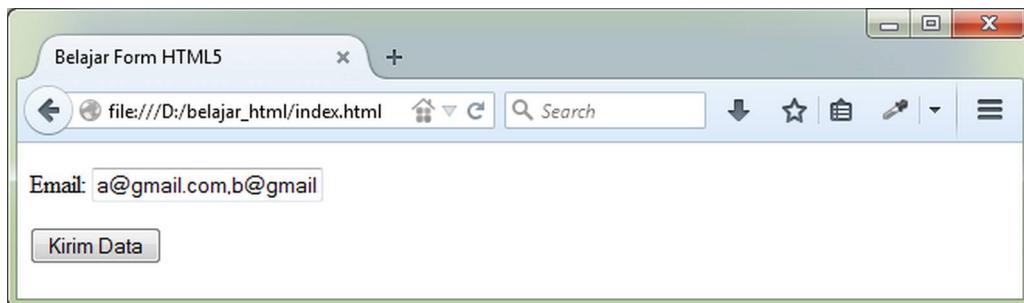
Jika email sesuai dengan format, maka form akan dikirim ke server:

```
file:///D:/belajar_html/prosesform.php?email=duniailkom%40gmail.com
```

Tag **input type="email"** juga mendukung atribut **multiple**. Atribut ini bisa digunakan untuk men-input beberapa alamat email yang dipisah dengan tanda koma:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Email: <input type="email" name="email" multiple></p>
10  <p><input type="submit" value="Kirim Data"></p>
11 </form>
12 </body>
13 </html>
```



Gambar: Tampilan form `input type="email"` dengan atribut `multiple`

Selain validasi, `input type="email"` juga memudahkan proses input menggunakan *mobile web browser*, seperti **Android**. Ketika pengisian form, keyboard virtual android akan langsung menyediakan tombol '@' agar mudah dipilih.

Pesan Error Saat Validasi

Pesan error saat yang ditampilkan saat proses validasi berasal dari web browser. Jika web browser di-set menggunakan **Bahasa Indonesia**, pesan error ini juga akan ditampilkan dalam bahasa Indonesia. Jika settingan menggunakan **Bahasa Inggris** (default) maka pesan error ditampilkan dalam bahasa Inggris.

Untuk mengubah isi pesan ini, bisa dilakukan melalui JavaScript.



Gambar: Tampilan pesan kesalahan di Opera yang di set dengan Bahasa Indonesia

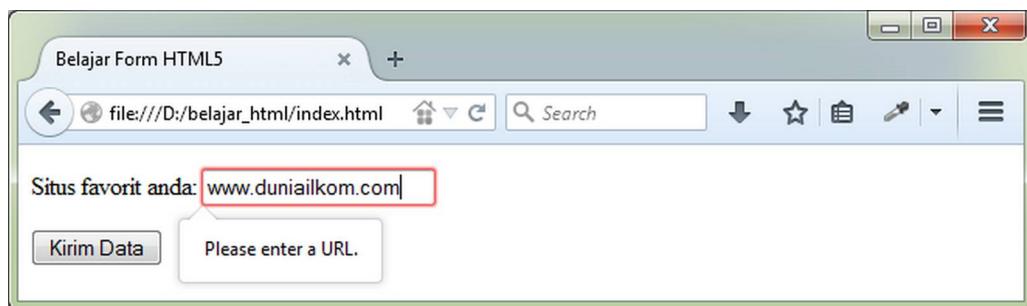
12.4 Input Element Type URL

Tag `<input>` dengan atribut `type="url"` menyediakan kotak input dengan proses validasi URL (alamat web). Secara garis besar, prosesnya mirip dengan `input type email`, yakni akan menampilkan pesan error jika pengguna menginput sesuatu yang bukan alamat URL. Berikut contoh penulisannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Situs favorit anda: <input type="url" name="situs"></p>
10  <p><input type="submit" value="Kirim Data"></p>
11 </form>
12 </body>
13 </html>
```

Perlu dicatat juga bahwa alamat URL ini harus ditulis bagian *protocol*-nya, yakni bagian: `http://`. Jika tidak, pesan error tetap tampil. Sebagai contoh, jika yang diinput alamat adalah: `www.duniaikom.com`, alamat tersebut dianggap tidak valid, kita harus menulisnya dengan lengkap: `http://www.duniaikom.com`.



Gambar: Tampilan form `input type="url"`

Ketika objek form URL dibuka dengan mobile web browser (android/iOS), keyboard virtual akan menyembunyikan tombol spasi (yang tidak pernah digunakan pada URL), dan menggantinya dengan tombol seperti tanda titik (.), backslash (\) dan “.com”. Ini agar penulisan alamat URL menjadi lebih efisien.

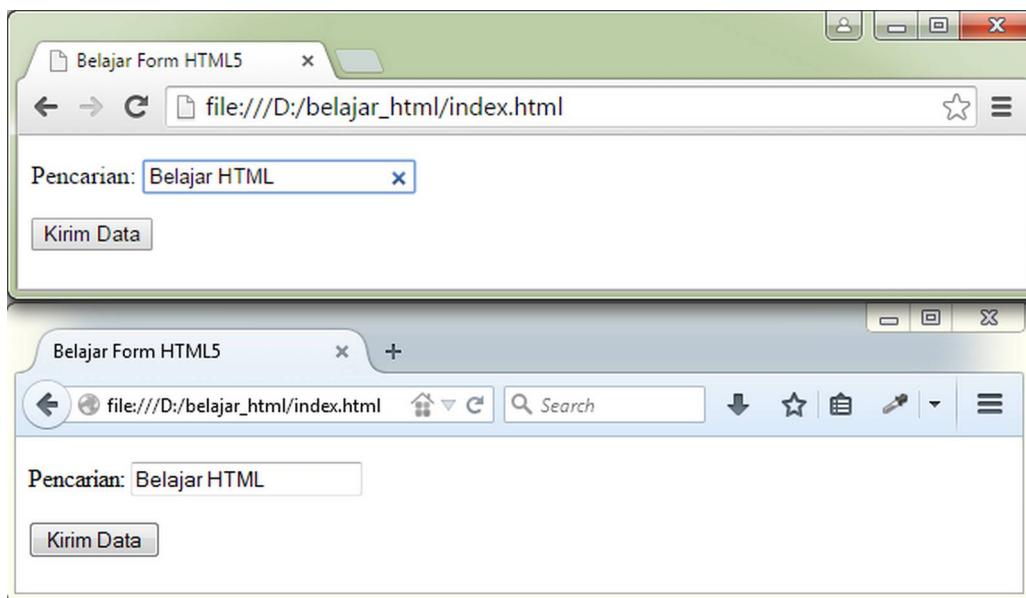
12.5 Input Element Type Search

Hampir setiap web memiliki fitur pencarian. Karena hal inilah HTML5 juga menyediakan atribut `type="search"` untuk tag `<input>`. Tampilan `input type search` tidak banyak perubahan, bahkan di dalam Mozilla Firefox hanya terlihat text box biasa.

Berikut contoh penggunaan tag input type search:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Pencarian: <input type="search" name="cari"></p>
10  <p><input type="submit" value="Kirim Data"></p>
11 </form>
12 </body>
13 </html>
```



Gambar: Tampilan form `input type="search"` pada Google Chrome (atas) dan Mozilla Firefox (bawah)

Seperti yang terlihat di dalam web browser Mozilla Firefox, tidak ada perubahan apa-apa, namun jika dijalankan pada Google Chrome dan Opera, akan terlihat sedikit tambahan tampilan, yakni terdapat tanda silang diakhir text box saat kita mulai mengetik sesuatu. Ini digunakan untuk menghapus teks yang telah ditulis.

12.6 Input Element Type Number

Jika kita memiliki inputan form yang hanya boleh diisi dengan angka, bisa menggunakan tag `<input>` dengan atribut `type="number"`. Penggunaan objek form ini akan mengaktifkan proses validasi jika karakter yang diinput bukan angka.

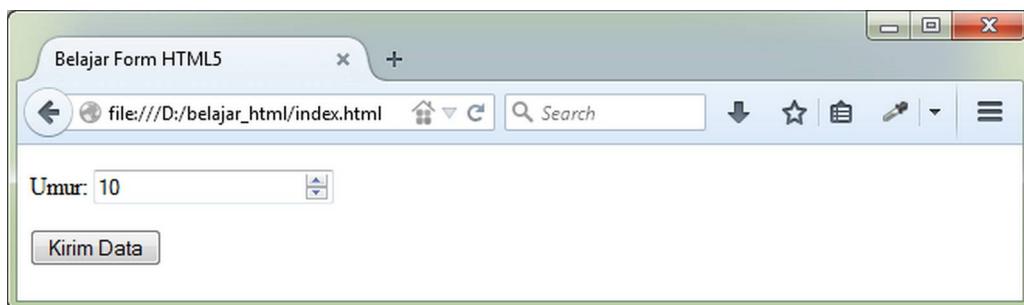
Selain itu, web browser juga menambahkan tombol **up** dan **down** di akhir text box. Tombol ini dikenal dengan sebutan **spin button**.

Berikut contoh penggunaan tag input type number:

index.html

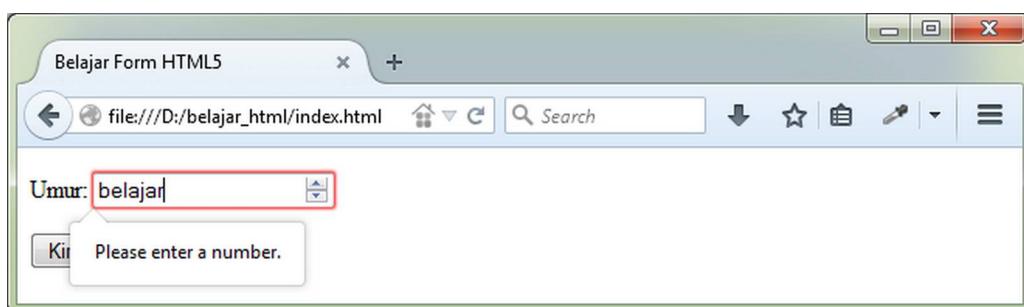
```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Umur: <input type="number" name="umur"></p>
10  <p><input type="submit" value="Kirim Data"></p>
11 </form>
```

```
12 </body>
13 </html>
```



Gambar: Tampilan form `input type="number"`

Jika pengguna menginput karakter selain angka, akan tampil pesan error dari web browser.



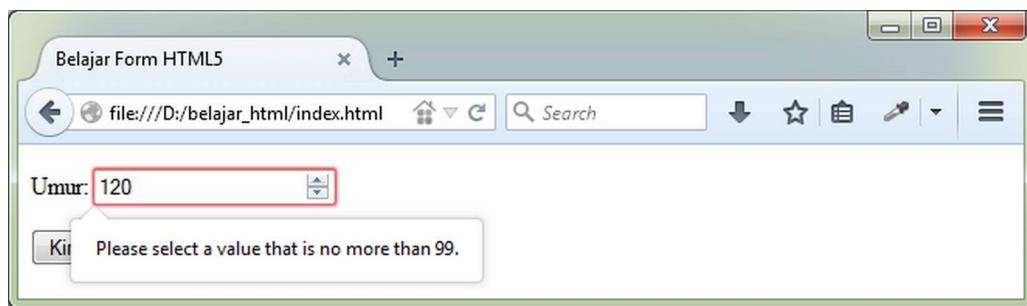
Gambar: Tampilan form `input type="number"` dengan pesan error

Atribut Min dan Max

Agar validasi lebih fokus, kita juga bisa menambahkan atribut **min** dan **max**. Sesuai dengan namanya, atribut ini membatasi angka minimum dan maksimum. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Umur: <input type="number" name="umur" min="0" max="99"></p>
10  <p><input type="submit" value="Kirim Data"></p>
11 </form>
12 </body>
13 </html>
```



Gambar: Tampilan form `input type="number"` dengan atribut `min` dan `max`

Kali ini selain karakter angka, web browser juga akan menampilkan error jika angka yang diinput melebihi nilai `min` dan `max`. Kedua atribut ini juga bisa digunakan untuk objek form lainnya, seperti `date`.

Atribut Step

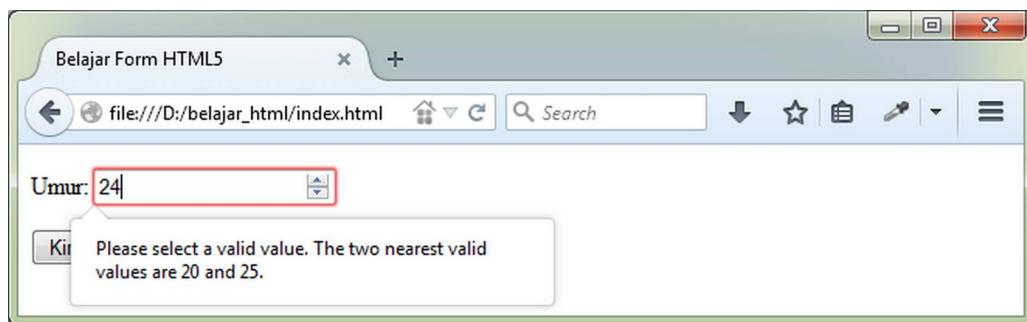
Satu lagi atribut yang bisa kita tambahkan untuk number adalah `step`. Atribut ini digunakan untuk menaikkan/menurunkan nilai angka menggunakan *spin button* di sisi kanan text box. Selain itu, atribut ini juga membatasi angka yang bisa diinput.

Sebagai contoh, jika saya membuat atribut `step="5"`, maka angka yang bisa diinput hanyalah kelipatan 5. Di luar itu, web browser akan menampilkan pesan error:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Umur: <input type="number" name="umur" min="0" max="99" step="5"></p>
10  <p><input type="submit" value="Kirim Data"></p>
11 </form>
12 </body>
13 </html>
```

Dalam contoh di atas pengguna hanya bisa menginput umur dalam kelipatan 5, sedikit pemakaian memang :)

Gambar: Tampilan form `input type="number"` dengan atribut `step`

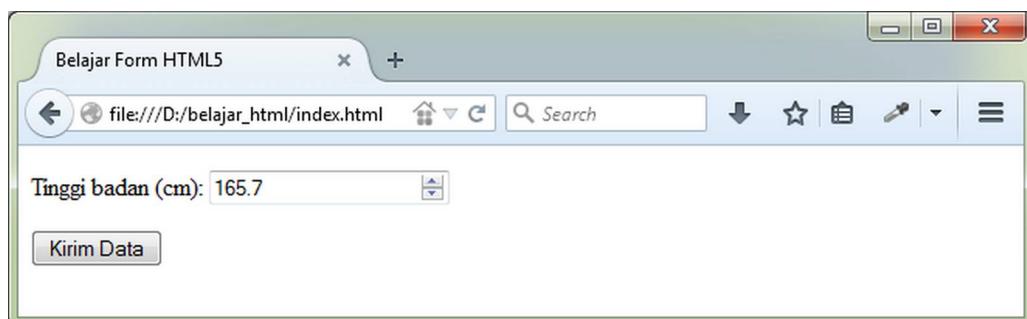
Walaupun '1.2' adalah angka, namun tag `input type number` tetap menampilkan pesan error. Ini terjadi karena secara default web browser menggunakan nilai `step="1"`, sehingga angka desimal tidak bisa diinput.

Solusinya, kita bisa menuliskan atribut `step="0.1"`, dan kali ini angka desimal seperti '0.4', '99.9' akan lolos validasi:

index.html

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Form HTML5</title>
6   </head>
7   <body>
8     <form action="prosesform.php" method="get">
9       <p>Tinggi badan (cm): <input type="number" name="umur" min="0"
10      step="0.1"></p>
11      <p><input type="submit" value="Kirim Data"></p>
12    </form>
13  </body>
14 </html>
```

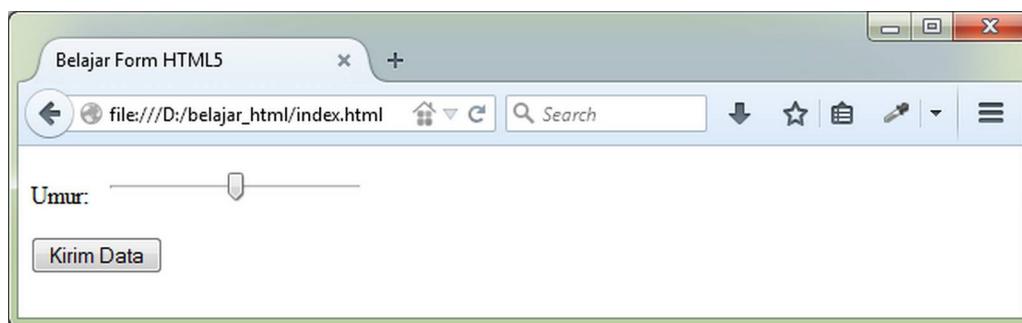
Gambar: Tampilan form `input type="number"` dengan atribut `step="0.1"`

12.7 Input Element Type Range

Tag <input> dengan atribut type="range" akan menampilkan slider untuk menginput angka. Tampilan ini akan memudahkan pengguna karena tidak perlu mengetik teks, tapi cukup menggeser slider saja. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Umur: <input type="range" name="umur" min="0" max="99" step="5"></p>
10  <p><input type="submit" value="Kirim Data"></p>
11 </form>
12 </body>
13 </html>
```



Gambar: Tampilan form `input type="range"`

Akan tetapi seperti yang terlihat, objek form ini tidak menampilkan angka yang dipilih, sangat tidak nyaman digunakan karena kita tidak tahu berapa nilainya.

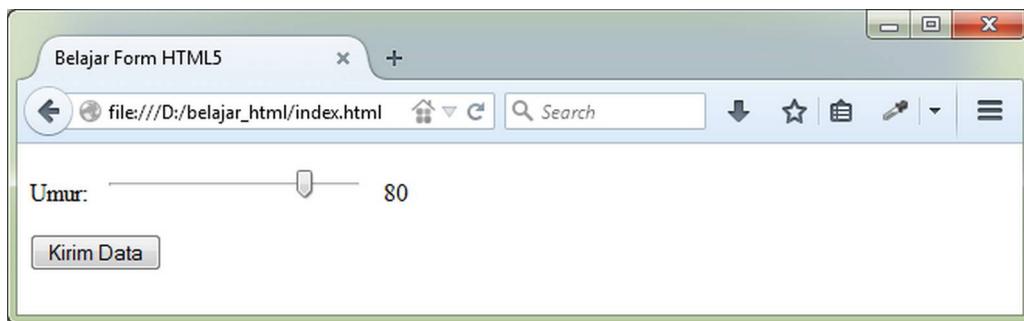
Agar angka dari slider dapat ditampilkan, kita harus meminta bantuan JavaScript, seperti contoh berikut:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Umur: <input type="range" name="umur" min="0" max="99" step="5"
10  onchange="updateNilaiSlider(this.value);">
11   <span id="nilai_slider"></span></p>
12   <p><input type="submit" value="Kirim Data"></p>
13 </form>
14 <script>
15   function updateNilaiSlider(val) {
16     document.getElementById('nilai_slider').innerHTML=val;
17   }
18 </script>

```



Gambar: Tampilan form `input type="range"` dengan bantuan JavaScript

Kali ini, nilai dari slider ditampilkan ke dalam tag ``.

12.8 Input Element Type Tel

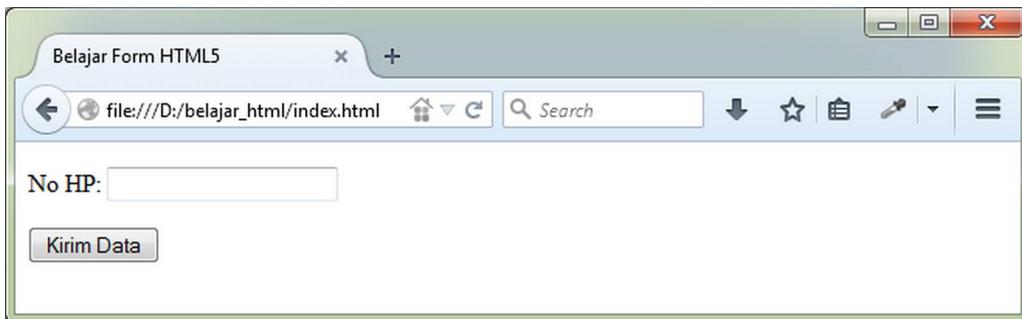
Atribut `type="tel"` pada tag `<input>` dirancang untuk inputan no telpon, baik berupa *fixed line* (no telp rumah yang menggunakan kode area) atau no HP. Karena banyaknya jenis format penulisan no telp, HTML5 tidak mengatur validasi untuk `input type tel`.

Jadi untuk apa fungsi atribut ini? Web browser mobile seperti android bisa memanfaatkan atribut `type="tel"` untuk mengatur tampilan keyboard virtual. Sebagai contoh, ketika pengguna mengakses bagian form ini, keyboard akan beralih ke mode angka, sehingga memudahkan pengisian form.

Berikut contoh penggunaan tag `input type tel`:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>No HP: <input type="tel" name="telp"></p>
10  <p><input type="submit" value="Kirim Data"></p>
11 </form>
12 </body>
13 </html>
```

Gambar: Tampilan form `input type="tel"`

12.9 Input Element Type Date, Datetime, Datetime-local, Time, Month, dan Week

Membuat objek form untuk penginputan tanggal bukanlah sesuatu yang mudah. Seperti yang pernah kita lakukan di akhir bab sebelum ini, membuat tag `<select>` untuk pemilihan tanggal memerlukan 3 bagian terpisah, dan hasilnya juga tidak terlalu menarik.

Karena itulah HTML5 menyediakan objek form untuk penginputan **tanggal** (*date*), dan **waktu** (*time*). Tidak hanya 1, tetapi 6 objek form sekaligus, yakni **date**, **time**, **datetime-local**, **datetime**, **month** dan **week**. Semuanya digunakan sebagai nilai atribut type untuk tag `<input>`.

Berikut adalah perbedaan ke-6 type input ini:

- **Date:** Input berupa tanggal dengan format: YYYY-MM-DD, contohnya 25 april 2015 akan menjadi 2015-04-25.
- **Time:** Input berupa waktu dengan format: HH:MM, sebagai contoh jam 3:15 siang akan menjadi 15:15.

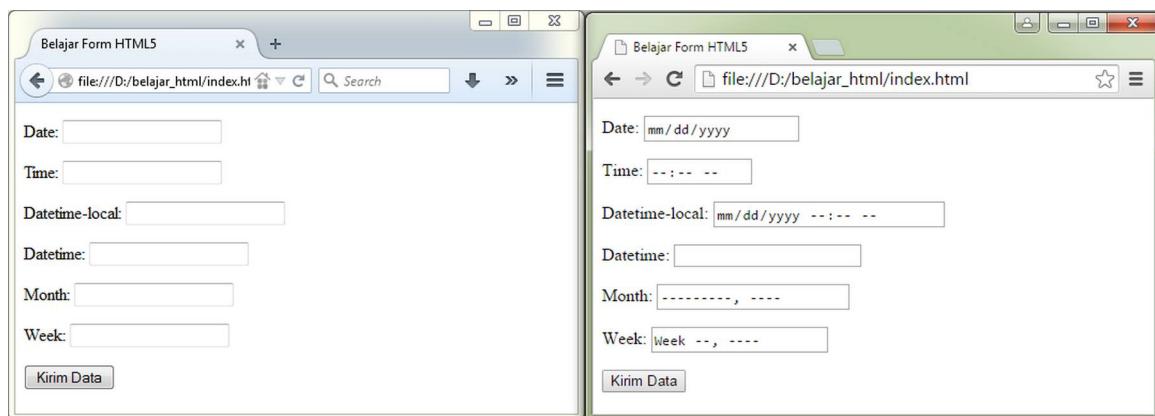
- **Datetime-local:** Input berupa tanggal dan waktu dengan format YYYY-MM-DDTHH:mm). Perhatikan bahwa tanda T memisahkan antara tanggal dan waktu. Sebagai contoh, 25 april 2015 pukul 3:15 siang akan menjadi 2015-04-15T15:15.
- **Datetime:** Input berupa tanggal dan waktu dengan format YYYY-MM-DD HH:mm:ss-HH:mm. Format ini menggunakan *time-zone offset*, akan tetapi datetime kebanyakan tidak didukung web browser, sebaiknya gunakan datetime-local.
- **Month:** Input berupa bulan dan tahun dengan format YYYY-MM, sebagai contoh bulan April 2015 akan menjadi: 2015-04.
- **Week:** Input berupa urutan minggu dalam 1 tahun, dengan format YYYY-Www. Nilai maksimal dari week bisa bernilai 52 atau 53, tergantung tahun. Sebagai contoh, Minggu keempat pada tahun 2015 akan menjadi 2015-W04.

Pada saat penulisan buku ini, web browser Mozilla Firefox 35 belum mendukung tag **input type date**. Agar bisa dijalankan, gunakan Google Chrome atau Opera.

Berikut adalah kode HTML yang menampilkan seluruh tag input type date:

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Form HTML5</title>
6   </head>
7   <body>
8     <form action="prosesform.php" method="get">
9       <p>Date: <input type="date" name="date"></p>
10      <p>Time: <input type="time" name="time"></p>
11      <p>Datetime-local: <input type="datetime-local" name="datetime-local"></p>
12      <p>Datetime: <input type="datetime" name="datetime"></p>
13      <p>Month: <input type="month" name="month"></p>
14      <p>Week: <input type="week" name="week"></p>
15      <p><input type="submit" value="Kirim Data"></p>
16    </form>
17  </body>
18 </html>
```



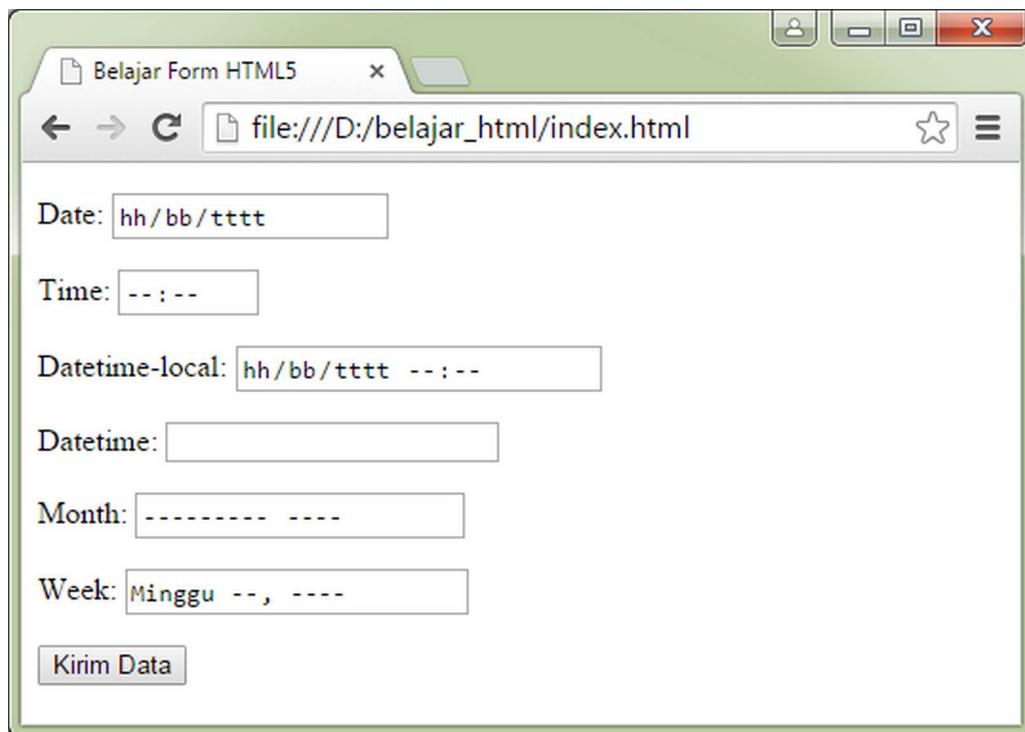
Gambar: Tampilan form untuk objek tanggal pada Mozilla Firefox (kiri) dan Google Chrome (kanan)

Seperti yang terlihat, web browser Mozilla Firefox hanya menampilkan text box biasa, tanpa proses validasi, sedangkan Google Chrome akan menampilkan kotak input tanggal (kecuali type **datetime** yang tidak didukung).

Tampilan di atas mungkin berbeda dengan yang terlihat di web browser anda. Ini karena web browser akan menyesuaikannya dengan settingan bahasa yang digunakan.

Sebagai contoh, jika Google Chrome di-set menggunakan bahasa Indonesia, kotak inputan tanggal akan ditampilkan dengan format DD-MM-YYYY, serta menggunakan nama hari dan bulan dalam bahasa Indonesia.

Namun apabila Google Chrome disetting menggunakan bahasa inggris (Amerika), format tanggal yang dipakai adalah dalam bentuk MM-DD-YYYY.



Gambar: Tampilan form untuk objek tanggal Google Chrome, dengan settingan Bahasa Indonesia

Terlepas dari format tampilan web browser, ketika form diinput, format yang dikirim sesuai dengan list pada halaman sebelum ini. Dimana **input type date** akan dikirim dengan format YYYY-MM-DD. Hal ini memudahkan kita mengolah data tersebut disisi server.

Berikut adalah hasil yang didapat ketika form di submit:

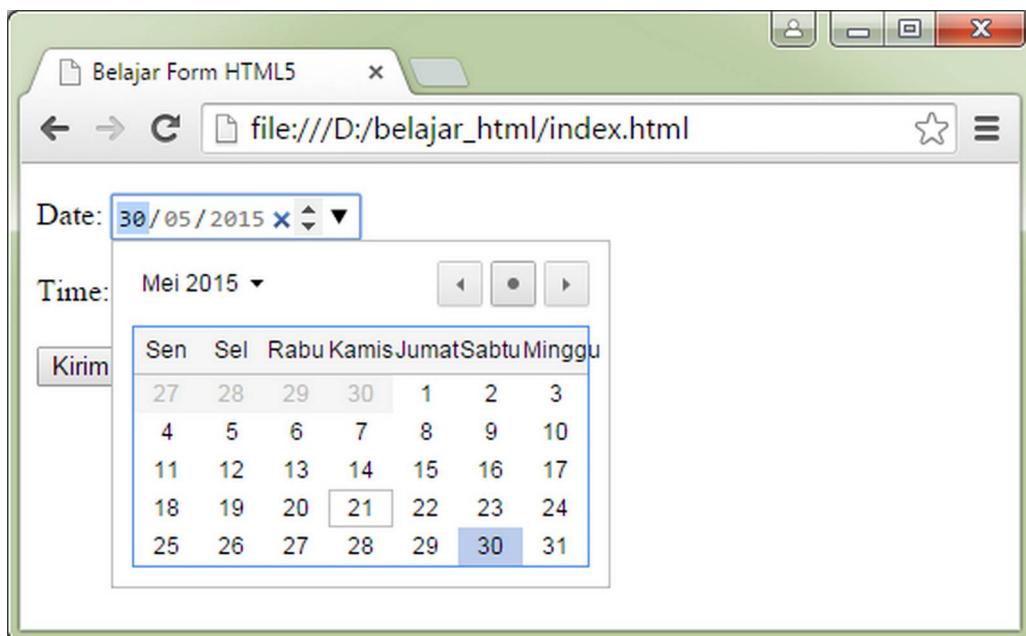
```
file: //D:/belajar_html/prosesform.php?date=2015-04-30&time=22%3A02
&datetime-local=2015-04-18T11%3A55&datetime=&month=2015-04&week=2015-W51
```

Masing-masing objek form dikirim sesuai dengan format masing-masing. Karakter %3A adalah karakter ‘:’ yang dikonversi ke dalam bentuk ASCII.

Tag input type date juga bisa ditambahkan atribut **min** dan **max** untuk membatasi range tanggal, berikut contohnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Date: <input type="date" name="date" min="2015-05-01"
10    max="2015-05-31"></p>
11   <p>Time: <input type="time" name="time" min="08:00" max="15:00"></p>
12   <p><input type="submit" value="Kirim Data"></p>
13 </form>
14 </body>
15 </html>
```



Gambar: Tampilan form `input type="date"` dengan atribut `min` dan `max`

Kotak inputan **date** di atas hanya bisa diisi dengan tanggal di bulan mei 2015 saja, dan untuk inputan **time** hanya bisa diisi dengan nilai dari pukul 8 hingga pukul 15.



Tag input type date ini sangat menarik untuk digunakan, namun karena belum didukung oleh Firefox, ada baiknya anda mempertimbangkan hal ini.

12.10 Atribut Autofocus

Biasanya, pada saat halaman yang berisi form ditampilkan, kita mulai dengan men-klik objek form paling atas kemudian mengisi form secara berurutan ke bawah. Akan lebih *user friendly* jika cursor teks langsung berada pada objek form pertama tanpa menunggu pengguna men-klik form. Fitur inilah yang bisa didapat jika menggunakan atribut **autofocus**.

Dengan memberikan atribut **autofocus** pada objek form pertama, pengguna bisa langsung mulai mengisi form tanpa harus men-kliknya terlebih dahulu. Berikut contohnya:

index.html

```

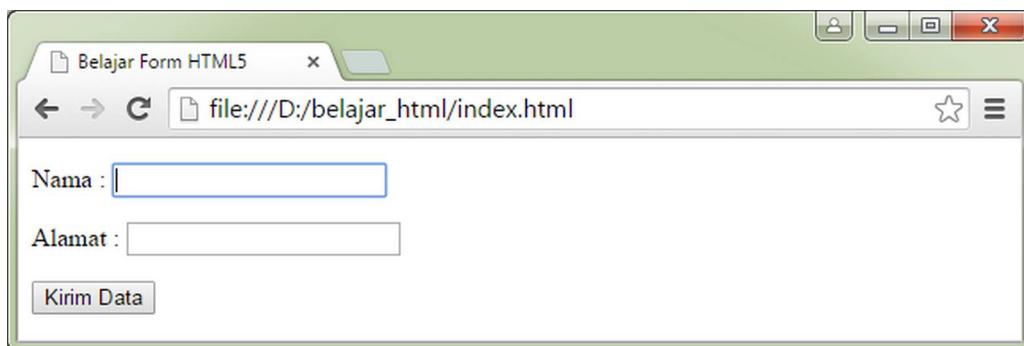
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="nama_user" autofocus></p>

```

```

10 <p>Alamat : <input type="text" name="alamat_user"></p>
11 <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>

```



Gambar: Tampilan form dengan atribut autofocus

Ketika form ditampilkan, cursor teks akan langsung berada pada inputan Nama.

12.11 Atribut Placeholder

Atribut **placeholder** digunakan untuk membuat teks sementara (*placeholder*) di dalam objek form. Teks ini akan hilang saat pengguna mulai mengetik sesuatu, sehingga cocok digunakan untuk memberi contoh cara pengisian form.

Berikut contoh penggunaan atribut **placeholder**:

index.html

```

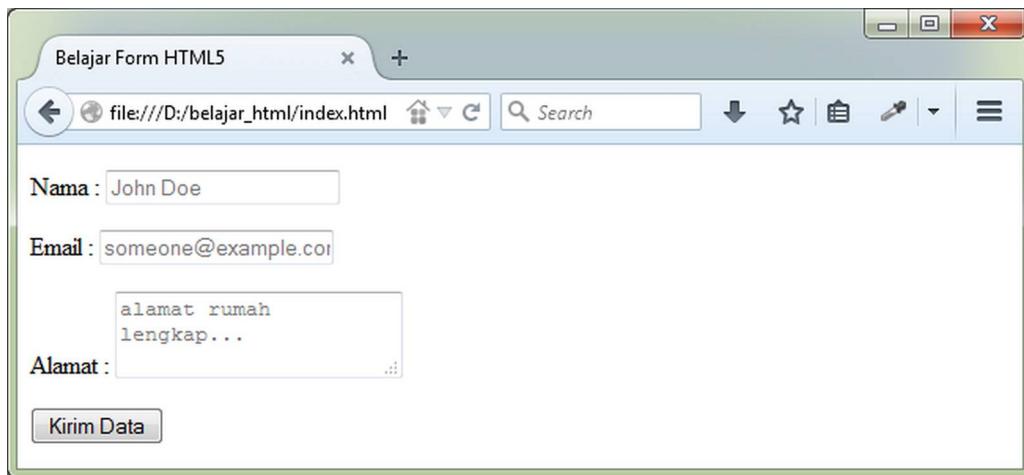
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Form HTML5</title>
6   </head>
7   <body>
8     <form action="prosesform.php" method="get">
9       <p>Nama :<br/>
10      <input type="text" name="nama_user" placeholder="John Doe">
11    </p>
12    <p>Email :<br/>
13      <input type="email" name="email_user" placeholder="someone@example.com">
14    </p>
15    <p>Alamat :<br/>
16      <textarea name="alamat" placeholder="alamat rumah lengkap..."></textarea>
17    </p>

```

```

18  <p><input type="submit" value="Kirim Data"></p>
19 </form>
20 </body>
21 </html>

```



Gambar: Tampilan form dengan atribut placeholder

Berbeda dengan atribut **value**, nilai dari **placeholder** tidak akan dikirim untuk diproses. Jika pengguna tidak mengubah isian objek form, maka form dianggap dalam keadaan kosong. Sebelum ada atribut ini, kita harus menggunakan JavaScript untuk mendapatkan efek yang sama.

12.12 Atribut Autocomplete

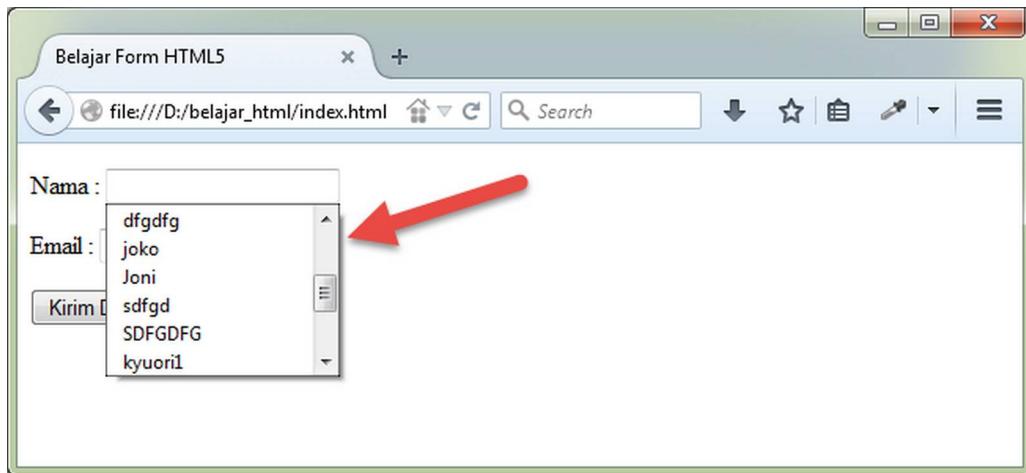
Secara default, web browser menyimpan banyak history mengenai cara kita berselancar, dan salah satu diantaranya adalah mengingat beberapa kata yang diinput melalui form. Sebagai contoh, jika kita membuat form seperti berikut ini:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="username"></p>
10  <p>Email : <input type="email" name="email"></p>
11  <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>

```



Gambar: Tampilan form dengan atribut autocomplete

Besar kemungkinan web browser akan menyediakan beberapa pilihan di kedua kotak input ini. Kata-kata tersebut berasal dari history web browser. Jika kita tidak ingin ini ditampilkan, tambahkan atribut `autocomplete="off"` pada tag `<input>`:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="username" autocomplete="off"></p>
10  <p>Email : <input type="email" name="email" autocomplete="off"></p>
11  <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>

```

Kali ini web browser tidak akan mengambil history ketika akan mengisi form. Selain nilai "off", kita juga bisa memberikan nilai `autocomplete="on"`, namun hal ini tidak perlu karena nilai "on" merupakan nilai default.

Selain meletakkan atribut `autocomplete` pada tag `<input>`, kita juga bisa membuatnya di dalam tag form. Dengan demikian seluruh objek form tidak akan menggunakan history web browser.

12.13 Atribut Required

Ketika kita membuat objek form yang memiliki fitur validasi seperti email, proses validasi form akan aktif jika inputan belum sesuai dengan format email. Akan tetapi jika kita mengosongkan input ini, form tetap terkirim tanpa pesan error.

Untuk ‘memaksakan’ pengguna untuk mengisi sebuah objek form, kita bisa menambahkan atribut **required** ke dalam tag tersebut. Berikut contohnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nama : <input type="text" name="username" required></p>
10  <p>Email : <input type="email" name="email" required></p>
11  <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>
```



Gambar: Tampilan form dengan atribut required

Kali ini, jika salah satu atau kedua kotak input tidak diisi, web browser akan menampilkan pesan error.

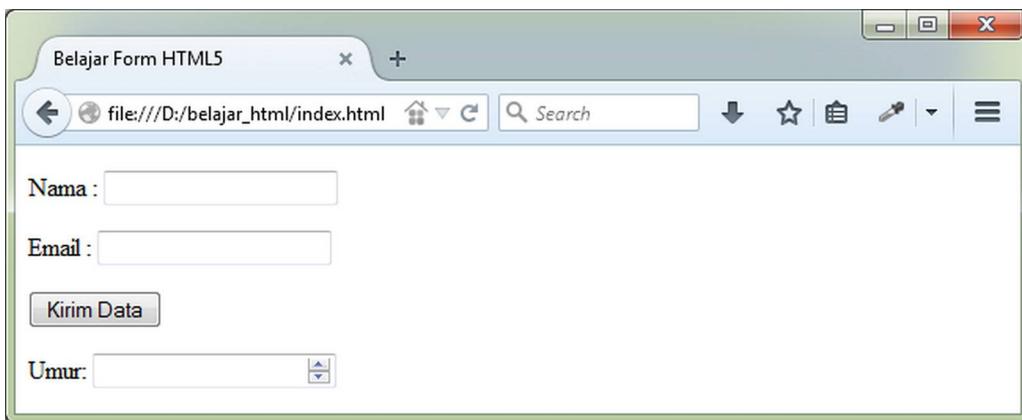
12.14 Atribut Form

Salah satu atribut yang cukup unik adalah atribut **form**. Atribut ini memungkinkan kita membuat objek form diluar tag `<form>`, tetapi tetap berada di dalam form. Langsung saja kita lihat contoh kode HTMLnya:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get" id="form_registrasi">
9   <p>Nama : <input type="text" name="username"></p>
10  <p>Email : <input type="email" name="email"></p>
11  <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 <p>Umur: <input type="number" name="umur" form="form_registrasi"></p>
14 </body>
15 </html>
```



Gambar: Tampilan form dengan atribut form

Dalam kode di atas, inputan **umur** berada di luar tag **<form>**, tetapi ketika form disubmit tag input ini ‘ikut’ dikirim bersama form.

Agar bisa terhubung, kita harus mengisi atribut **form** dengan nilai yang sama dengan atribut **id** pada tag **<form>**. Karena pada contoh di atas saya membuat atribut **id="form_registrasi"** maka di dalam tag input umur harus ditulis **form="form_registrasi"**.

12.15 Datalist Element dan Atribut List

Tag **<datalist>** dan pasangannya atribut **list** membawa fitur yang sangat menarik ke dalam form. Dengan keduanya, kita bisa membuat *list suggestion* untuk tag input.

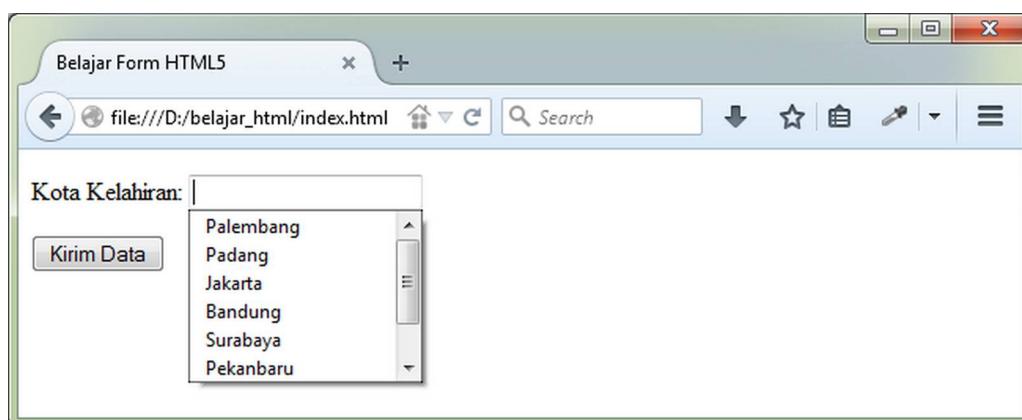
Sebagai contoh, ketika kita mengetik beberapa huruf pada kotak pencarian google, akan keluar beberapa kata *suggestion* atau saran dari google mengenai topik yang akan dicari. Dengan menggunakan HTML5, kita bisa membuat fitur yang sama.

Agar lebih mudah dipahami, berikut contoh penggunaan tag **<datalist>** dengan atribut **list**:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <datalist id="list_kota">
9   <option label="Palembang" value="palembang">
10  <option label="Padang" value="padang">
11  <option label="Jakarta" value="jakarta">
12  <option label="Bandung" value="bandung">
13  <option label="Surabaya" value="surabaya">
14  <option label="Pekanbaru" value="pekanbaru">
15  <option label="Makassar" value="makassar">
16  <option label="Medan" value="medan">
17 </datalist>
18 <form action="prosesform.php" method="get" id="form_registrasi">
19   <p>Kota Kelahiran: <input type="text" name="kota" list="list_kota"></p>
20   <p><input type="submit" value="Kirim Data"></p>
21 </form>
22 </body>
23 </html>
```



Gambar: Tampilan form dengan datalist

Dapat anda perhatikan, *list suggestion* ini pada dasarnya merupakan perpaduan antara tag `<select>` dengan tag `<input>`. Pada saat mulai mengetik beberapa huruf (atau double klik kotak text box), akan muncul daftar kata yang bisa dipilih.

Berbeda dengan tag `<select>` yang mengharuskan pengguna untuk memilih dari daftar yang tersedia, list suggestion ini kita bisa memilih kata yang direkomendasikan atau menggunakan kata baru (yang tidak ada dalam daftar).

Tag `<datalist>` digunakan untuk membuat daftar kata. Tag ini bisa berada dimana saja, tidak harus ditulis sebelum form, bisa juga diletakkan sesudah form, atau di akhir dokumen.

Mirip seperti tag `<select>`, daftar kata dibuat menggunakan tag `<option>`. Atribut **label** digunakan untuk membuat tampilan pilihan, sedangkan atribut **value** untuk nilai yang dikirim ketika form di submit. Beberapa web browser menampilkan *list suggestion* ini dengan sedikit perbedaan.

Agar tag input dapat terhubung dengan datalist, nilai dari atribut **list** harus sama dengan **id** pada tag `<datalist>`. Dalam contoh di atas, atribut `list="list_kota"` pada tag `<input>` sesuai dengan atribut `id="list_kota"` pada tag `<datalist>`.

12.16 Atribut Pattern

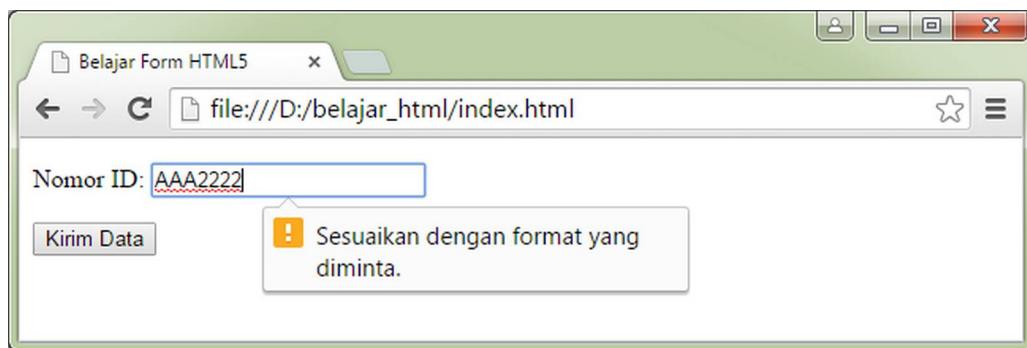
Atribut **pattern** digunakan untuk membuat aturan validasi yang kita rancang sendiri dengan *regular expression*.

Regular expression (sering disingkat dengan **regex** atau **regexp**) adalah kumpulan karakter yang digunakan untuk pencocokan pola. *Regular Expression* adalah topik yang cukup luas dan tidak akan kita bahas disini.

Sebagai contoh, jika kita ingin membuat text box yang hanya bisa diisi dengan 3 digit huruf besar kemudian diikuti dengan 3 digit angka, maka regexnya adalah: “[A-Z]{3}-[0-9]{3}”. Berikut contoh kode HTML penggunaan atribut pattern tersebut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Nomor ID: <input type="text" name="no_id"
10    placeholder="AAA111" pattern="[A-Z]{3}[0-9]{3}"></p>
11   <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>
```



Gambar: Tampilan form dengan atribut pattern (error karena pola tidak sesuai)

Jika kita menginput Nomor ID tidak sesuai dengan pola, akan tampil pesan error.

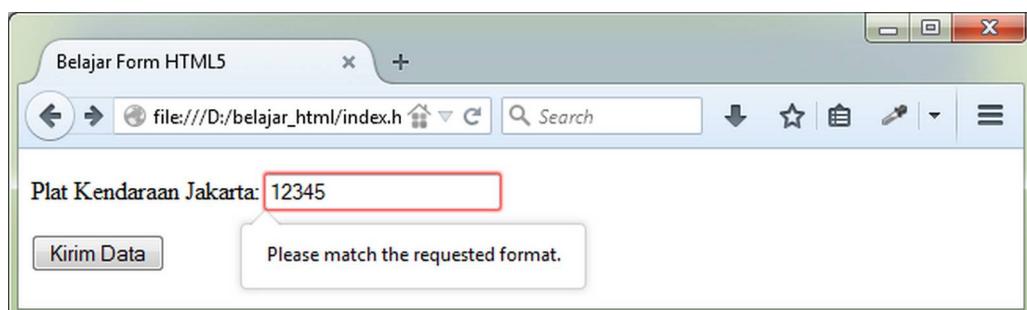
Sebagai contoh lain, jika kita ingin membuat pola nomor plat kendaraan Jakarta yang diawali dengan huruf B, kemudian diikuti dengan angka, dan diakhiri dengan beberapa karakter huruf, bisa menggunakan regex: “[B] [0-9]+[A-Z]+”. Berikut contohnya:

index.html

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Form HTML5</title>
6   </head>
7   <body>
8     <form action="prosesform.php" method="get">
9       <p>Plat Kendaraan Jakarta: <input type="text"
10      name="no_id" placeholder="B1111XX" pattern="[B][0-9]+[A-Z]+></p>
11      <p><input type="submit" value="Kirim Data"></p>
12    </form>
13  </body>
14 </html>
```

Penggunaan atribut **pattern** ini sangat fleksibel, karena kita bisa membuat pola validasi yang ditentukan sendiri.



Gambar: Tampilan form dengan atribut pattern (error karena pola tidak sesuai)



Saya menyarankan anda untuk mulai mempelajari Regular Expression. Karena hampir setiap bahasa pemrograman modern mendukung regex, seperti JavaScript dan PHP. Regex hampir selalu digunakan dalam proses validasi.

12.17 Atribut Novalidate

Beberapa objek form seperti `email`, `number` dan `url` memiliki validasi bawaan. Walaupun validasi sangat berguna, ada kemungkinan dimana kita ingin me-nonaktifkan proses validasi ini. Untuk keperluan tersebut bisa menggunakan atribut `novalidate`. Atribut ini ditempatkan di dalam tag `<form>`. Berikut contohnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get" novalidate>
9   <p>Email: <input type="email" name="email"></p>
10  <p>URL: <input type="url" name="url"></p>
11  <p><input type="submit" value="Kirim Data"></p>
12 </form>
13 </body>
14 </html>
```

Form di atas akan melewati proses validasi karena atribut `novalidate` pada tag `<form>`.

12.18 Atribut Formaction, Formenctype, Formmethod dan Formtarget

Atribut `formaction`, `formenctype`, `formmethod` dan `formtarget` berfungsi untuk menimpa atribut serupa yang didefinisikan dalam tag `<form>`, yaitu `atribut action`, `enctype`, `method`, dan `target`. Keempat atribut ini hanya bisa digunakan pada tag `input type submit` dan `input type image`.

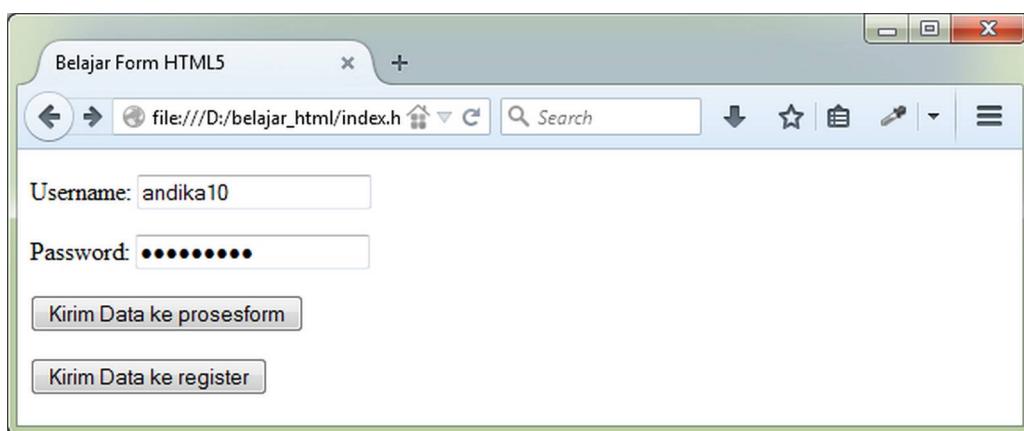
Contoh penerapannya adalah untuk membuat form yang bisa mengirim hasil ke alamat yang berbeda. Perhatikan contoh berikut:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Username: <input type="text" name="username"></p>
10  <p>Password: <input type="password" name="pass"></p>
11  <p><input type="submit" value="Kirim Data ke prosesform"></p>
12  <p><input type="submit" value="Kirim Data ke register"
13    formaction="register.php"></p>
14 </form>
15 </body>
16 </html>

```



Gambar: Tampilan form dengan atribut formaction

Pada form di atas, terdapat 2 buah tombol **submit**. Jika tombol “Kirim Data ke prosesform” ditekan, form akan dikirim ke halaman **prosesform.php**, yakni sesuai dengan atribut **action="prosesform.php"** yang tertulis pada tag **<form>**.

Apabila yang di klik adalah tombol “Kirim Data ke register”, form akan dikirim ke halaman **register.php**, sesuai dengan atribut **formaction="register.php"** yang ada di dalam tag submit ini.

Mirip dengan **formaction**, atribut **formenctype**, **formmethod** dan **formtarget** juga berfungsi sama, yakni mendefinisikan ulang atribut yang ada di dalam tag **<form>**. Berikut contoh lainnya:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get">
9   <p>Username: <input type="text" name="username"></p>
10  <p>Password: <input type="password" name="pass"></p>
11  <p><input type="submit" value="Kirim Data ke prosesform"></p>
12  <p><input type="submit" value="Kirim Data ke register"
13    formaction="register.php" formmethod="post"
14    formtarget="_blank" formenctype="multipart/form-data"></p>
15 </form>
16 </body>
17 </html>

```

Kali ini jika tombol “Kirim data ke register” di klik, form akan dikirim ke halaman `register.php` menggunakan method *post*, encoding *multipart/form-data* dan tampil di jendela yang baru.

12.19 Progress Element

Walaupun tag `<progress>` bukanlah bagian dari form (karena tidak meminta inputan), tag ini sering dibahas bersamaan dengan form. Begitu juga dengan beberapa tag dan atribut yang akan kita bahas setelah ini.

Tag `<progress>` digunakan untuk menampilkan garis proses (*progress bar*) yang mirip seperti kita men-copy file di dalam Windows. Efek tampilannya berbeda-beda tergantung web browser yang digunakan.

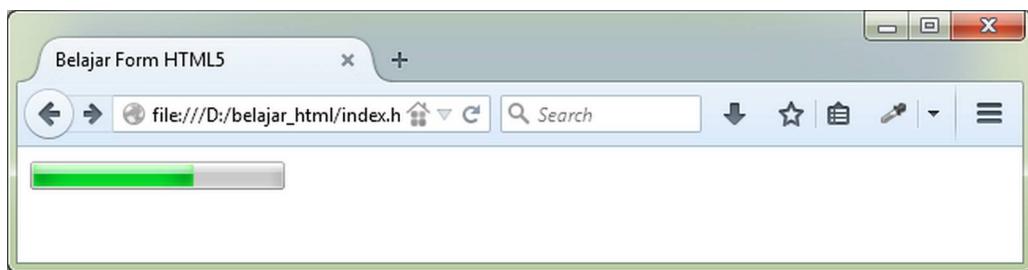
Berikut contoh penggunaan tag `<progress>`:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8   <progress value="0.65"></progress>
9 </body>
10 </html>

```



Gambar: Tampilan tag <progress>

Atribut **value** berfungsi untuk mengatur seberapa besar proses yang telah terjadi. Kode di atas akan menampilkan garis proses yang ‘penuh’ 65%, ini karena secara default nilai minimum adalah 0 (progress bar kosong) dan nilai maksimal 1 (progress bar penuh).

Untuk mengubah skala ini, bisa menggunakan atribut **max**, seperti contoh berikut:

```
<progress value="0.65"></progress>
```

Kali ini garis progress akan ditampilkan sebesar 50% (posisi nilai 250 dari angka maksimal 500).

Agar lebih informatif, kita bisa menambahkan keterangan tambahan yang berisi nilai progress bar saat ini:

```
<progress value="250" max="500"></progress>
```

Progress bar cocok digunakan untuk menampilkan durasi suatu proses, seperti berapa lama proses upload berlangsung, akan berapa lama proses input ke database. Untuk hal ini kita harus mengkombinasikannya dengan JavaScript.

12.20 Meter Element

Secara visual, meter element sangat mirip dengan progress element. Selain itu tag <meter> juga mendukung atribut yang sama. Tag <meter> lebih ditujukan untuk menampilkan jangkauan nilai seperti tinggi badan, berat badan, dll. Berikut contoh penulisannya:

index.html

```

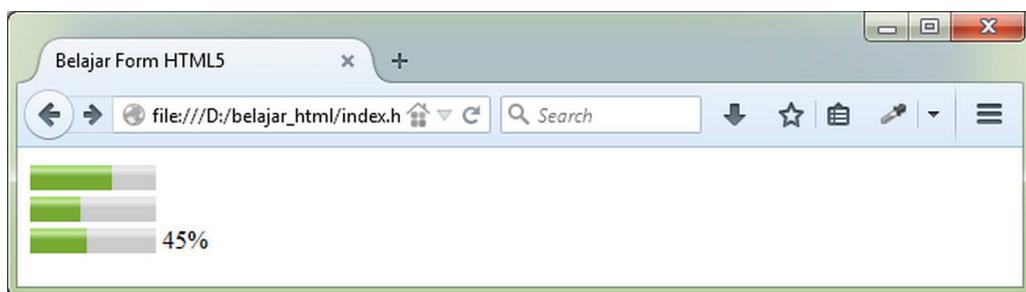
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Belajar Form HTML5</title>
6  </head>
7  <body>
8      <meter value="0.65"></meter>
9      <br>
10     <meter value="60" max="150"></meter>

```

```

11   <br>
12   <meter value="0.45"></meter><span> 45%</span>
13 </body>
14 </html>

```



Gambar: Tampilan tag <meter>

Tag <meter> ini cocok digunakan untuk membuat grafik tabulasi nilai.

12.21 Output Element

Sesuai dengan namanya, tag <output> digunakan sebagai penampung nilai keluaran (hasil) dari sebuah perhitungan. Karena HTML bukanlah bahasa pemrograman, tag ini hanya berguna jika digunakan bersama-sama dengan JavaScript atau PHP.

Sebelum di tambahkan di dalam HTML5, untuk penampung hasil ini biasanya menggunakan tag ‘generik’ seperti <div> atau . Contoh sederhana untuk tag <output> adalah untuk menampung hasil program kalkulator, seperti berikut ini:

index.html

```

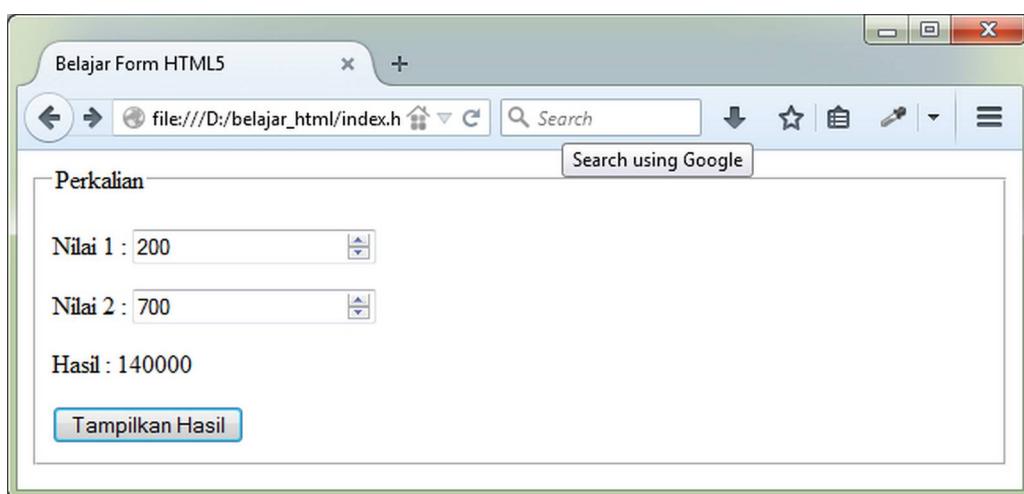
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8 <form action="prosesform.php" method="get" onsubmit="return false">
9 <fieldset>
10 <legend>Perkalian</legend>
11   <p>Nilai 1 : <input type="number" id="nilai_1"></p>
12   <p>Nilai 2 : <input type="number" id="nilai_2"></p>
13   <p>Hasil : <output id="hasil"></output></p>
14   <button onclick="hitung()">Tampilkan Hasil</button>
15 <script>
16   function hitung() {
17     var angka1=document.getElementById('nilai_1').value;

```

```

18     var angka2=document.getElementById('nilai_2').value;
19     var hasil= angka1*angka2;
20     document.getElementById('hasil').innerHTML=hasil;
21     return false;
22 }
23 </script>
24 </fieldset>
25 </form>
26 </body>
27 </html>

```



Gambar: Contoh penggunaan tag <output>

Dalam kode di atas saya menggunakan JavaScript untuk melakukan proses perhitungan, dan menampilkan hasilnya ke dalam tag <output>. Secara default, tag <output> ditampilkan seperti teks biasa, tanda efek tampilan tambahan.

Untuk proses seperti ini sebenarnya kita tidak harus menggunakan tag <output>, tag-tag lain juga akan berfungsi sama, namun tag <output> paling pas untuk tujuan ini.

12.22 Keygen Element

Sama seperti tag <output>, tag <keygen> juga hanya berfungsi jika dikombinasikan dengan JavaScript atau PHP. Tag <keygen> ditujukan untuk menampung nilai kunci (key) yang bisa berfungsi untuk proses enkripsi data, pengecekan form, dll. Umumnya nilai key ini dihasilkan secara otomatis dari kode program PHP.

Contoh penggunaannya seperti berikut:

```
<keygen name="name" challenge="challenge string" keytype="type"
keyparams="pqg-params">
```

Pada update HTML 5.2, tag <keygen> dinyatakan *deprecated* dan sebaiknya tidak dipakai lagi.

12.23 Atribut Contenteditable

Tim penyusun HTML5 (W3C dan WHATWG) banyak mengambil tag dan atribut tidak standar yang potensial dan memasukkannya ke dalam standar HTML5. Atribut **contenteditable** adalah salah satunya. Atribut ini diperkenalkan sejak Internet Explorer 5 namun tidak menjadi standar HTML.

Atribut **contenteditable** menyediakan fitur untuk membuat text editor sederhana di dalam web browser. Ketika sebuah element ditambah atribut `contenteditable="true"`, kita bisa mengedit teks tersebut. Langsung saja kita lihat contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8   <p contenteditable="true">Anda bisa mengedit text ini</p>
9 </body>
10 </html>
```



Gambar: Contoh penggunaan atribut `contenteditable="true"`

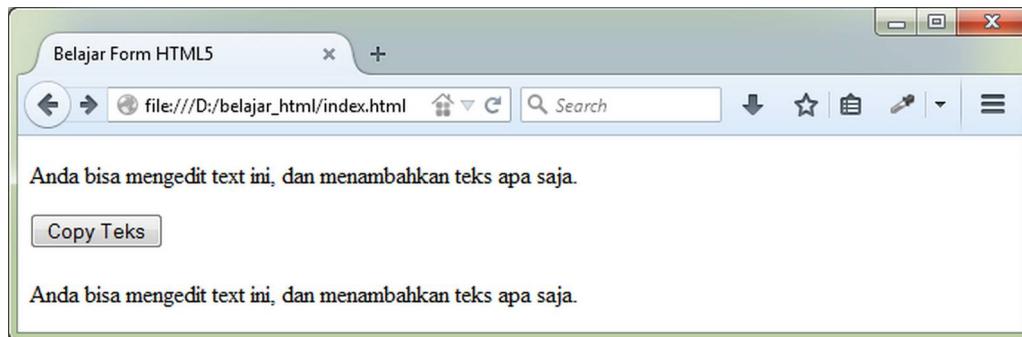
Silahkan jalankan kode di atas, dan klik teks menggunakan mouse. Cursor mouse akan berubah menjadi tanda “|” yang menandakan teks bisa diedit.

Fitur seperti ini cocok dikombinasikan dengan JavaScript, sehingga teks bisa disimpan atau digunakan untuk keperluan lain.

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8   <p contenteditable="true" id="editor">Anda bisa mengedit text ini</p>
9   <button onclick="getText()">Copy Teks</button>
10  <br><br>
11  <div id="container"></div>
12
13 <script>
14   function getText(){
15     var a=document.getElementById("editor").innerHTML;
16     document.getElementById("container").innerHTML=a;
17   }
18 </script>
19 </body>
20 </html>
```



Gambar: Contoh penggunaan atribut `contenteditable="true"` dengan JavaScript

Dalam kode di atas, saya menggunakan JavaScript untuk mengambil hasil teks yang sudah diedit.

12.24 Atribut Designmode

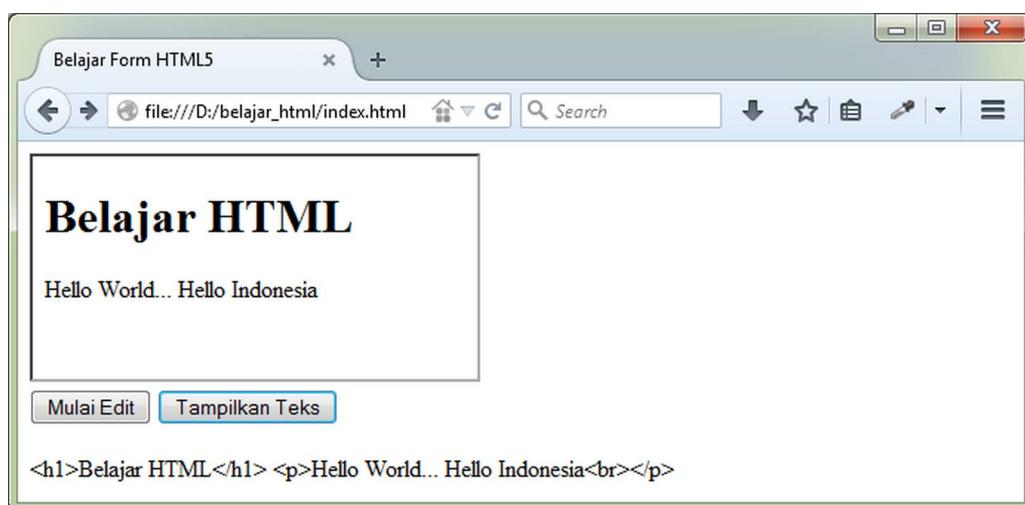
Atribut designmode mirip dengan contenteditable, dimana kita bisa mengedit teks di dalam web browser.

Jika contenteditable mengubah sebuah element (tag) menjadi teks editor, atribut designmode akan membuat seluruh halaman menjadi editor. Karena itu, atribut ini umumnya di set menggunakan JavaScript dan ditempatkan di dalam `<iframe>` (cara penggunaan tag `<iframe>` akan kita bahas dalam bab tersendiri).

Berikut contoh penggunaan atribut designmode:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Form HTML5</title>
6 </head>
7 <body>
8   <iframe id="editor" src="hello_world.html"></iframe>
9   <br>
10  <button onclick="startEdit()">Mulai Edit</button>
11  <button onclick="getText()">Tampilkan Teks</button>
12  <br><br>
13  <div id="container"></div>
14
15 <script>
16   function startEdit(){
17     var editor=document.getElementById("editor");
18     editor.contentWindow.document.designMode = "on";
19   }
20   function getText(){
21     document.getElementById("container").textContent=
22       editor.contentWindow.document.body.innerHTML;
23   }
24 </script>
25 </body>
26 </html>
```



Gambar: Contoh penggunaan atribut `designMode="on"` dengan JavaScript

Dalam kode di atas, saya menempatkan halaman test.html di dalam iframe, kemudian mengubah atribut `designMode="on"` ketika tombol **"Mulai Edit"** di-klik, sehingga kita bisa mengedit teks di

dalam **iframe**.

Selanjutnya, tombol “Tampilkan Teks” digunakan untuk mengambil teks dari **iframe**. Anda bisa berkreasi dengan menggunakan atribut ini dan membuat sebuah teks editor sederhana.

Dalam bab ini kita telah membahas berbagai tag dan atribut form yang dibawa oleh HTML5. Dengan menggabungkan materi yang dibahas pada bab sebelumnya, kita sudah membahas hampir seluruh tag dan atribut untuk membuat form di dalam HTML.

Berikutnya, akan dibahas tentang cara menginput aplikasi lain ke dalam halaman HTML menggunakan `<embed>`, `<object>` dan `<iframe>` element.

13. Embed, Object dan Iframe Element

Dalam bab ini kita akan mempelajari 3 tag HTML yang bisa digunakan untuk menginput file dari luar halaman HTML atau aplikasi lain ke dalam halaman web. Tiga tag tersebut adalah <embed>, <object>, dan <iframe>.

13.1 Embed Element

Tag <embed> digunakan untuk menampilkan aplikasi lain ke dalam halaman web. Aplikasi ini bisa berupa file multimedia seperti video, audio, animasi flash, dll. Untuk menjalankannya, di dalam web browser harus tersedia plugin yang sesuai, seperti *adobe flash player*.

Tag <embed> memiliki beberapa atribut. Atribut yang paling penting adalah **src**. Atribut **src** digunakan untuk menentukan lokasi dari file yang akan ditampilkan.

Selain itu, kita juga bisa menambahkan informasi tentang file yang diinput melalui atribut **type**. Atribut ini diisi dengan nilai MIME type. Jika atribut **type** tidak ditulis, web browser akan mencoba ‘menebak’ jenis file dari nama file yang di embed.

Untuk mengatur tinggi dan lebar tampilan, bisa ditambahkan atribut **width** dan **height**. Kedua atribut ini diisi dengan angka dalam satuan pixel.

Berikut beberapa contoh penulisan tag <embed>:

```
<embed src="animasi_flash.swf">
<embed src="sebuah_video.mp4" type="video/mp4">
<embed src="sebuah_video.flv" width="250" height="100">
<embed src="abc.pdf" type="application/pdf">
```

Walaupun tag <embed> sudah ada sejak lama dan didukung penuh oleh berbagai web browser, tag ini tidak termasuk ke dalam standar HTML4. Dengan kata lain, sebuah halaman yang menggunakan tag <embed> tidak akan lolos validasi HTML4.

Di dalam HTML5, tag <embed> akhirnya secara resmi diakui menjadi standar. Walaupun demikian, penggunaan tag <embed> saat ini tidak terlalu banyak, dan digantikan dengan tag <object> atau <iframe>.

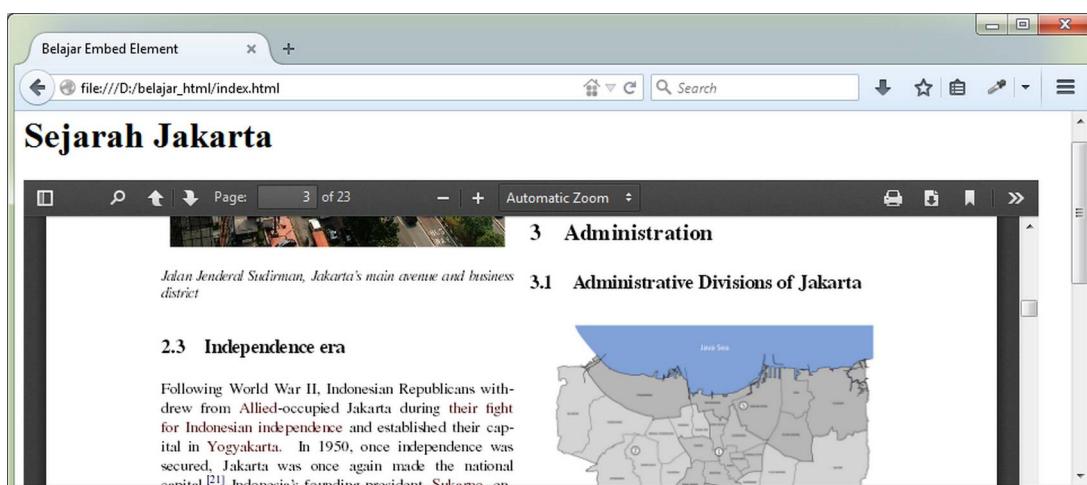
Sebagai contoh lain, berikut adalah tampilan tag <embed> yang digunakan untuk menampilkan sebuah file PDF langsung di halaman website:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Embed Element</title>
6 </head>
7 <body>
8   <h1>Sejarah Jakarta</h1>
9   <embed src="Jakarta.pdf" type="application/pdf" width="900" height="600">
10 </body>
11 </html>

```



Gambar: Tampilan file pdf dengan menggunakan tag `<embed>`

Tampilan di atas hanya bisa berjalan jika web browser memiliki plugin pdf (biasanya dapat ketika saat menginstall aplikasi PDF Reader).

Tag `<embed>` dan juga tag `<object>` adalah cara yang paling sering digunakan untuk menjalankan video menggunakan flash (sebelum era HTML5 yang membawa tag `<video>`).

13.2 Object Element

Fungsi dari tag `<object>` hampir sama dengan tag `<embed>`. Tag `<object>` juga digunakan untuk menambahkan sesuatu ke dalam halaman HTML.

Berbeda dengan tag `<embed>`, tag `<object>` menggunakan atribut `data` untuk mencari lokasi file yang akan ditampilkan. Selain itu, tag ini juga menggunakan tag penutup `</object>`.

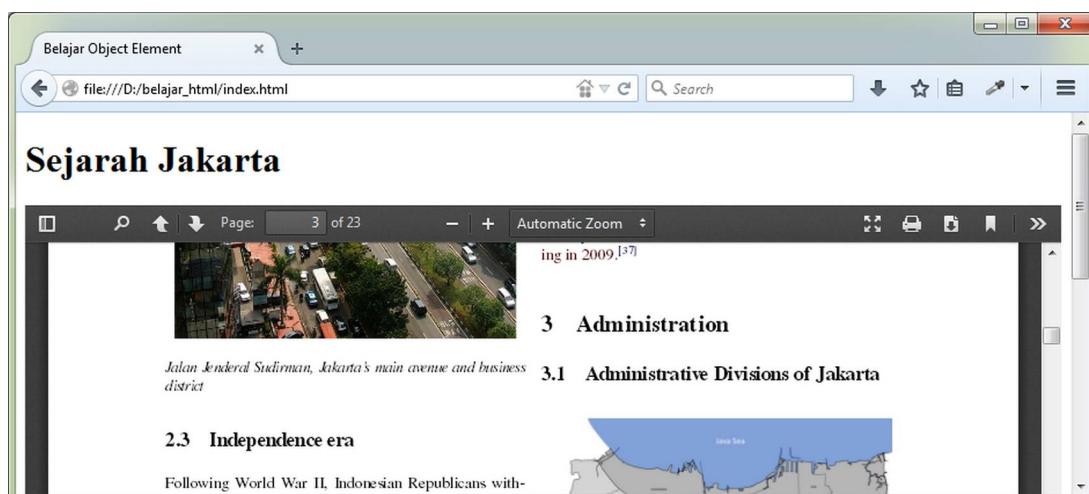
Berikut contoh penggunaannya dalam menampilkan file pdf:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Object Element</title>
6 </head>
7 <body>
8   <h1>Sejarah Jakarta</h1>
9   <object data="Jakarta.pdf" type="application/pdf" width="100%" height="600"></object>
10 </body>
11 </html>

```



Gambar: Tampilan file pdf dengan menggunakan tag <object>

Beberapa pendapat mengatakan bahwa tag <object> lebih baik daripada tag <embed>. Ini dikarenakan sejarah tag <embed> yang tidak secara resmi menjadi standar HTML 4. Selain itu, kita juga bisa menampilkan sesuatu untuk web browser tidak mendukung tag <object> (mempersiapkan *fallback*). Berikut contohnya:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Object Element</title>
6 </head>
7 <body>
8   <h1>Sejarah Jakarta</h1>
9   <object data="Jakarta.pdf" type="application/pdf" width="100%" height="600">
10    Alternatif Download : <a href="m17apr.pdf">m17apr.pdf</a>

```

```
11  </object>
12 </body>
13 </html>
```

Kode di atas akan menampilkan halaman pdf langsung di web browser, dan akan menampilkan link download untuk web browser yang tidak mendukung tag `<object>`.

Walaupun relatif jarang digunakan, tag `<object>` juga bisa digunakan untuk menampilkan halaman HTML lain:

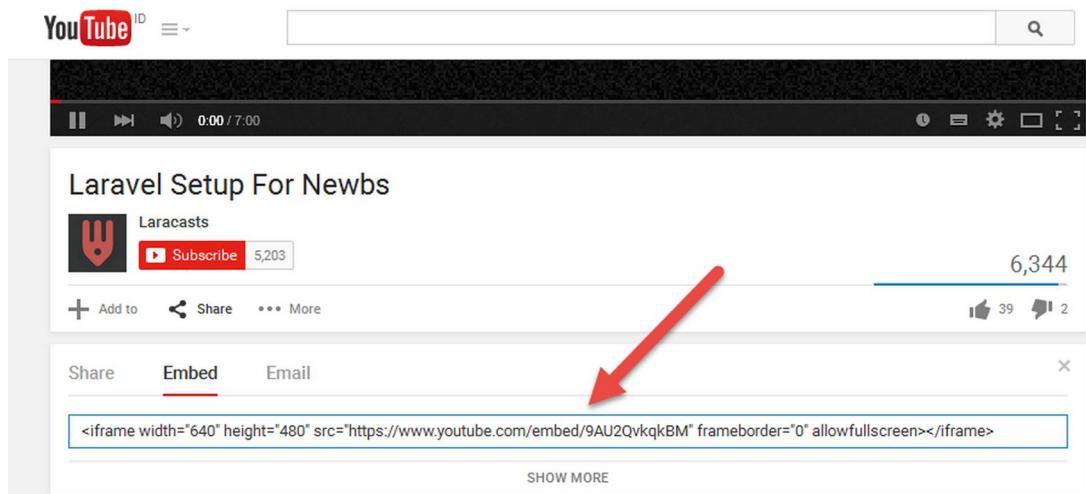
index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Object Element</title>
6 </head>
7 <body>
8   <object data="hello_world.html" type="text/html"
9     width="100%" height="600"></object>
10 </body>
11 </html>
```

Untuk menampilkan halaman HTML lain seperti ini, lebih sering menggunakan tag `<iframe>`.

13.3 Iframe Element

Tag `<iframe>` juga berfungsi untuk memasukkan aplikasi luar ke dalam halaman HTML. Saat ini tag `<iframe>` lebih sering digunakan daripada tag `<object>` dan `<embed>`. Sebagai contoh, jika anda ingin menampilkan video dari YouTube, YouTube akan memberikan kode dalam bentuk `<iframe>` (walaupun link dari youtube bernama ‘embed’). Berikut salah satu contoh tag `<iframe>` yang disediakan oleh YouTube:

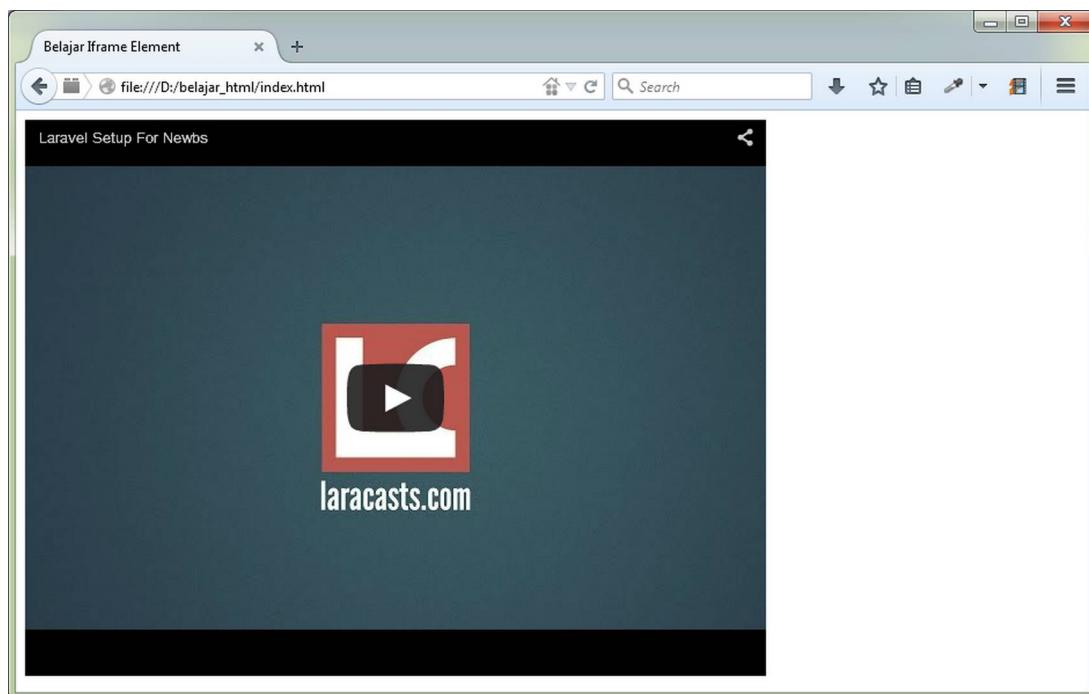


Gambar: Cara embed video dari youtube menggunakan tag <iframe>

Untuk menggunakannya, kita tinggal copy paste kode tersebut ke dalam HTML:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Iframe Element</title>
6 </head>
7 <body>
8   <iframe width="640" height="480"
9     src="https://www.youtube.com/embed/9AU2QvkqkBM" frameborder="0"
10    allowfullscreen></iframe>
11 </body>
12 </html>
```

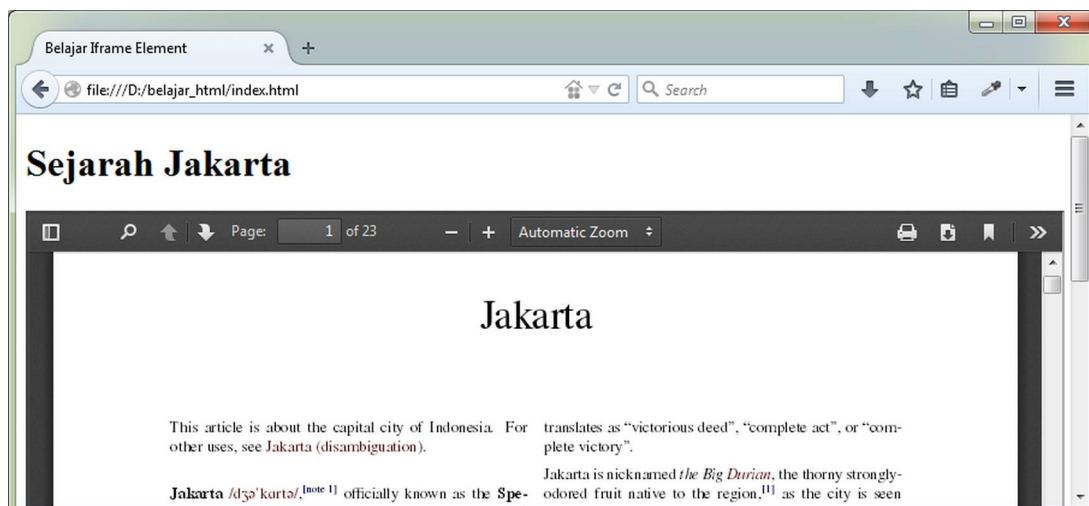


Gambar: Tampilan embed video dari youtube menggunakan tag <iframe>

Mirip dengan tag <embed>, tag <iframe> menggunakan atribut **src** untuk memasukkan file, atribut **width** untuk mengatur lebar dan atribut **height** untuk mengatur tinggi jendela. Seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Iframe Element</title>
6 </head>
7 <body>
8   <h1>Sejarah Jakarta</h1>
9   <iframe src="Jakarta.pdf" width="100%" height="600"></iframe>
10 </body>
11 </html>
```



Gambar: Tampilan file pdf dengan menggunakan tag <iframe>

Selain untuk memasukkan file multimedia seperti video, tag <iframe> banyak digunakan untuk menyisipkan halaman lain ke halaman saat ini, seperti contoh berikut:

index.html

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Belajar Iframe Element</title>
6  </head>
7  <body>
8      <iframe src="http://www.duniaikom.com" width="80%" height="300" scrolling="no"></iframe>
9  </body>
10 </html>

```

Dalam kode di atas, saya menampilkan isi dari situs duniaikom menggunakan tag <iframe>.



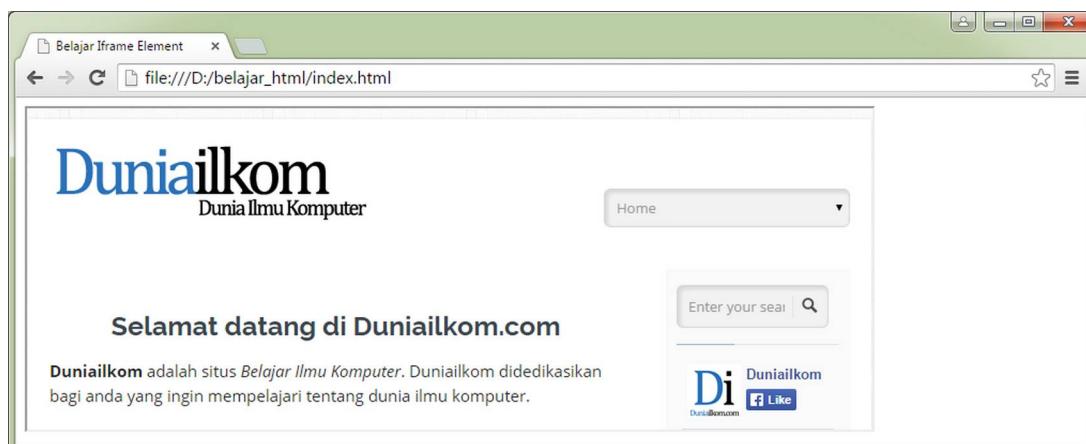
Gambar: Tampilan situs duniaikom dengan menggunakan tag <iframe>

Seperti yang terlihat, kita mendapatkan sebuah halaman HTML di dalam halaman HTML, lengkap dengan scrollbar. Jika anda ingin scrollbar ini tidak ditampilkan, tinggal tambahkan atribut scrolling="no":

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Iframe Element</title>
6 </head>
7 <body>
8   <iframe src="http://www.duniailkom.com" width="80%" 
9     height="300" scrolling="no"></iframe>
10 </body>
11 </html>
```



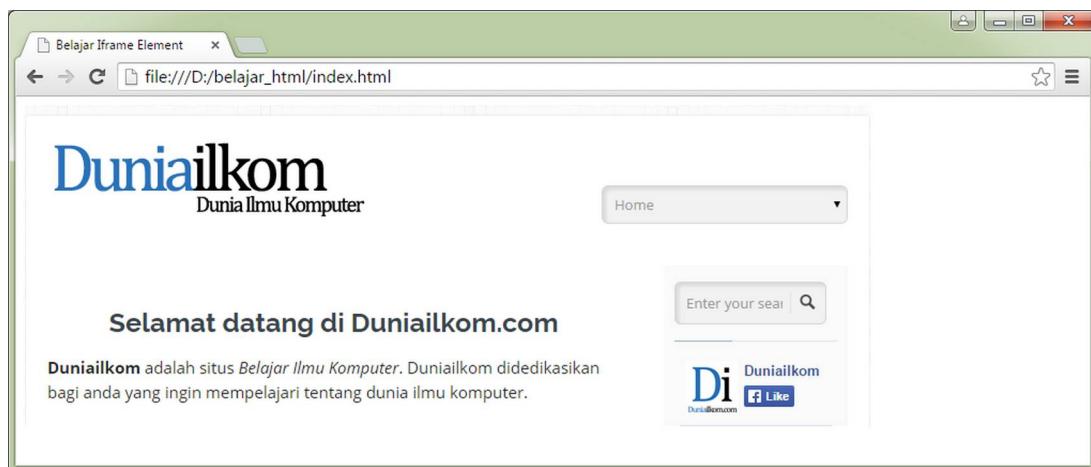
Gambar: Tampilan situs duniailkom dengan menggunakan tag <iframe> tanpa scroll

Untuk menghapus garis bingkai, bisa ditambahkan atribut frameborder="0":

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Iframe Element</title>
6 </head>
7 <body>
8   <iframe src="http://www.duniailkom.com" width="80%" 
9     height="300" scrolling="no" frameborder="0"></iframe>
10 </body>
11 </html>
```



Gambar: Tampilan situs duniaIlkom dengan menggunakan tag <iframe> tanpa border

- i** Beberapa penyedia layanan iklan juga menggunakan tag <iframe> untuk mamasukkan materi iklan ke website kita.

13.4 Pilih Embed, Object, atau iFrame?

Ketiga tag yang kita bahas dalam bab ini sama-sama digunakan untuk menampilkan aplikasi lain ke dalam halaman saat ini. Jadi mana yang terbaik?

Beberapa sumber banyak yang membahas hal ini, dan sebagian besar menyarankan menggunakan <iframe>. Kelebihan <iframe> jika dibandingkan dengan <embed> dan <object> adalah <iframe> bisa melakukan komunikasi dengan halaman ‘induk’ dimana tag ini ditampilkan.

Selain itu, bagi penyedia layanan seperti YouTube, dengan menggunakan tag <iframe>, YouTube bisa melakukan perubahan setting dan setiap halaman yang menggunakan <iframe> akan langsung terupdate, tanpa harus mengubah kode <iframe> sebelumnya.

Dalam bab ini kita telah membahas cara menginput aplikasi atau file lain ke dalam halaman HTML saat ini.

Berikutnya kita akan membahas salah satu fitur baru dalam HTML5: **Canvas Element**.

14. HTML5 Canvas Element

Beberapa waktu yang lalu, untuk dapat membuat gambar yang bisa tampil secara ‘real time’, kita harus menggunakan aplikasi ‘pihak ketiga’ seperti adobe flash atau microsoft silverlight.

Namun saat ini HTML5 membawa serta fitur canvas yang membolehkan kita membuat gambar langsung menggunakan HTML. Canvas membuka banyak kemungkinan yang sebelumnya bisa dibilang mustahil, seperti *web based game online*, *web based photo editing*, dll.

Walaupun demikian, canvas pada HTML5 sebenarnya bukan ‘murni’ HTML. Seperti yang akan anda pelajari, untuk ‘menggambar’ di dalam canvas, kita harus menggunakan JavaScript.



Karena buku ini ditujukan untuk pemula HTML, saya tidak akan membahas terlalu jauh seluruh fitur yang disediakan. Bab ini hanya sebagai pengantar HTML5 canvas. Ini karena penggunaan canvas harus menggunakan JavaScript.

14.1 Canvas Element

Untuk mulai menggambar, kita harus menyiapkan area yang akan menjadi ‘canvas’ dengan menggunakan tag <canvas>. Seperti contoh berikut:

```
<canvas id="kertas_gambar" width="500" height="300"></canvas>
```

Atribut **id** berfungsi sebagai ‘identitas’ atau ‘label’ dari tag <canvas>. Ini diperlukan karena kita akan mengaksesnya menggunakan JavaScript. Atribut **width** dan **height** digunakan untuk men-set tinggi serta lebar dari canvas, sehingga kode di atas akan membuat canvas dengan lebar 500 pixel dan tinggi 300 pixel.

Jika anda menempatkan kode tersebut ke dalam halaman **index.html**, berikut adalah kode HTML lengkapnya:

index.html

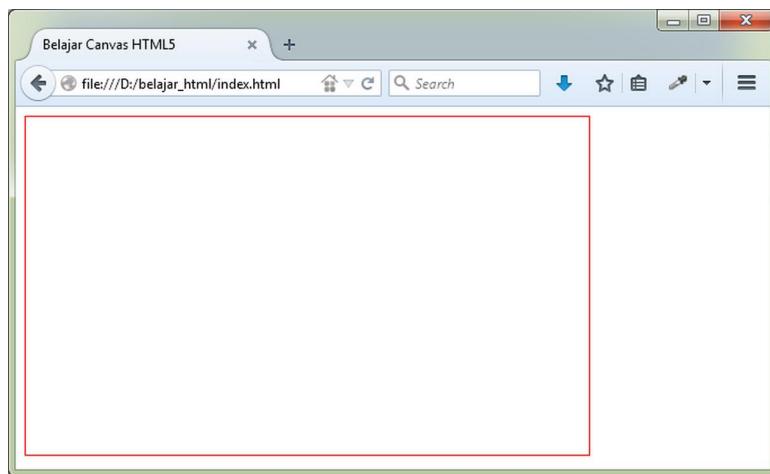
```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Canvas HTML5</title>
6 </head>
7 <body>
8   <canvas id="kertas_gambar" width="500" height="300"></canvas>
9 </body>
10 </html>
```

Jika anda menjalankan kode di atas, di dalam web browser tidak akan terlihat apa-apa. *Kenapa?* Ini karena secara default <canvas> tidak memiliki garis tepi. Jadi sebenarnya ‘*kertas_gambar*’ kita sudah tampil, tetapi tidak terlihat.

Agar lebih jelas, saya menambahkan sedikit CSS agar ‘*kertas_gambar*’ kita terlihat di web browser:

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Belajar Canvas HTML5</title>
6     <style>
7       canvas {
8         border: 1px solid red;
9       }
10    </style>
11  </head>
12  <body>
13    <canvas id="kertas_gambar" width="500" height="300"></canvas>
14  </body>
15 </html>
```



Gambar: Canvas siap untuk dilukis

Kali ini tag <canvas> akan ditampilkan dengan baris tepi berwarna merah sebesar 1 pixel. Ok, mari mulai menggambar... :)

14.2 Berkenalan dengan JavaScript untuk Canvas

Setelah ‘*kertas_gambar*’ kita tampil dengan sempurna di web browser, kode HTML yang dibutuhkan... sudah tidak ada lagi, dan saatnya masuk ke JavaScript.

Jika anda belum pernah menggunakan JavaScript, mungkin akan sedikit ‘terintimidasi’ dengan kode program yang akan kita gunakan. JavaScript sejak lama terkesan ‘angker’ bagi pemula web programming. Ini karena JavaScript adalah bahasa pemrograman murni, tidak seperti HTML dan CSS yang hanya ‘bahasa kode’. Ditambah lagi JavaScript menggunakan konsep pemrograman objek ..aaakk...

Walaupun demikian, saya akan mencoba menjelaskan baris per baris kode JavaScript yang digunakan. Apabila anda kurang paham, tidaklah menjadi masalah. Anda boleh mengabaikan maksud dari kode tersebut, dan hanya fokus melihat bagaimana hasil tampilannya. Setelah anda mempelajari JavaScript, kode-kode ini dapat dimengerti dengan mudah.

Untuk menggunakan JavaScript, kita harus menyisipkan tag `<script>` setelah penulisan tag `<canvas>`, seperti berikut ini:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Canvas HTML5</title>
6   <style>
7     canvas {
8       border: 1px solid red;
9     }
10  </style>
11 </head>
12 <body>
13   <canvas id="kertas_gambar" width="500" height="300"></canvas>
14   <script>
15
16   </script>
17 </body>
18 </html>
```

Posisi peletakan tag `<script>` sebenarnya bisa dimana saja, bisa dibagian `<head>`, atau pada bagian paling akhir halaman sebelum tag penutup `</body>` seperti contoh di atas.

Saya meletakkan tag `<script>` di posisi paling bawah halaman karena kita perlu mengakses tag `<canvas>` yang berada di atasnya. Ini disebabkan cara web browser yang memproses halaman web mulai dari baris paling atas, kemudian di proses baris-per-baris hingga baris paling bawah.

Apabila kita meletakkan tag `<script>` sebelum tag `<canvas>` (dibagian `<head>` seperti pada umumnya), maka kode JavaScript kita tidak bisa menemukan tag `<canvas>` yang berada di bawah (yang memang belum diproses oleh web browser).

Tapi saya pernah membaca bahwa kita bisa meletakkan tag `<script>` dibagian `<head>`, dan tetap bisa mengakses tag-tag HTML dibawahnya?

Yup, anda benar!, tetapi saya tidak akan membahas hal ini lebih jauh, karena kita sedang berada di buku tentang HTML, bukan JavaScript. Jika berminat, anda bisa mulai mempelajari JavaScript setelah membaca buku ini (sangat saya sarankan jika anda serius ingin menjadi web programmer).

Kembali ke canvas, untuk mulai ‘menggambar’ di tag `<canvas>`, kita harus mencari canvas mana yang ingin digambar. Di dalam JavaScript, hal ini dilakukan dengan perintah `document.getElementById()`.

Berikut adalah kode lengkapnya:

```
<script>
  var canvasku = document.getElementById("kertas_gambar");
</script>
```

Kode di atas akan mencari tag HTML yang memiliki `id="kertas_gambar"`, kemudian menyimpannya ke dalam variabel **canvasku**. Dengan kata lain, variabel **canvasku** akan berisi ‘objek’ HTML dengan `id="kertas_gambar"`.

Jika anda membuat tag `<canvas>` sebagai berikut:

```
<canvas id="kertasku" width="500" height="300"></canvas>
```

Maka kode JavaScriptnya akan menjadi:

```
<script>
  var canvasku = document.getElementById("kertasku ");
</script>
```

Langkah selanjutnya adalah menetapkan bahwa kita ingin menggambar 2 dimensi pada objek canvas ini. Perintah yang diperlukan adalah `getContext()`, seperti contoh berikut:

```
<script>
  var canvasku = document.getElementById("kertas_gambar");
  var context = canvasku.getContext("2d");
</script>
```

Sekarang, variabel **context** menjadi ‘objek penampung’ untuk seluruh perintah canvas yang akan kita tulis setelah ini. Anda bisa mengubah kata **context** menjadi kata lain.

14.3 Membuat Garis dalam Canvas

Untuk mulai membuat sebuah garis, tambahan 3 baris kode berikut ini setelah kode JavaScript sebelumnya:

```
context.moveTo(10,10);
context.lineTo(400,200);
context.stroke();
```

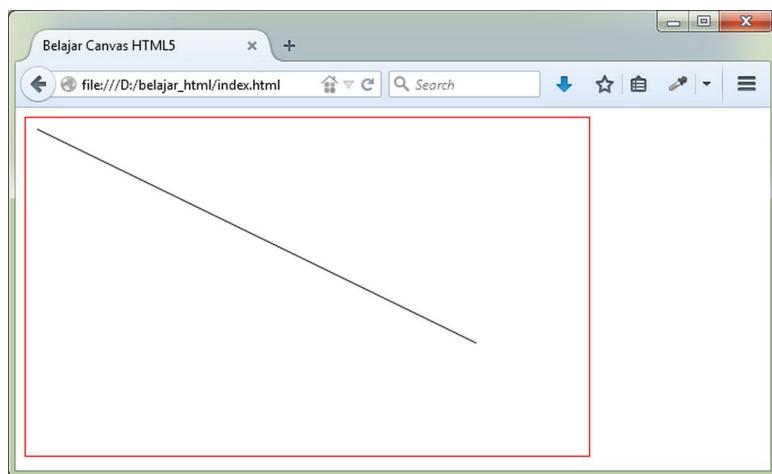
Kode di atas dapat diterjemahkan dengan “*buat garis mulai dari titik koordinat 10,10 sampai 400,200*”. Karena kita menggunakan ukuran canvas 500 pixel x 300 pixel, maka seharusnya akan terlihat sebuah garis dari sudut kiri atas ke sudut kiri bawah.

Kode program context.moveTo() digunakan untuk menandai koordinat awal garis. Perintah context.lineTo() digunakan untuk menandai koordinat akhir garis, dan perintah context.stroke() berguna untuk ‘menulis’ garis tersebut.

Berikut adalah seluruh kode HTMLnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Canvas HTML5</title>
6   <style>
7     canvas {
8       border: 1px solid red;
9     }
10  </style>
11 </head>
12 <body>
13  <canvas id="kertas_gambar" width="500" height="300"></canvas>
14  <script>
15    var canvasku = document.getElementById("kertas_gambar");
16    var context = canvasku.getContext("2d");
17    context.moveTo(10,10);
18    context.lineTo(400,200);
19    context.stroke();
20  </script>
21 </body>
22 </html>
```



Gambar: Canvas dengan sebuah garis

Selamat! Anda telah menguasai cara menggunakan canvas HTML5. Walaupun yang baru dihasilkan hanyalah sebuah garis sederhana.

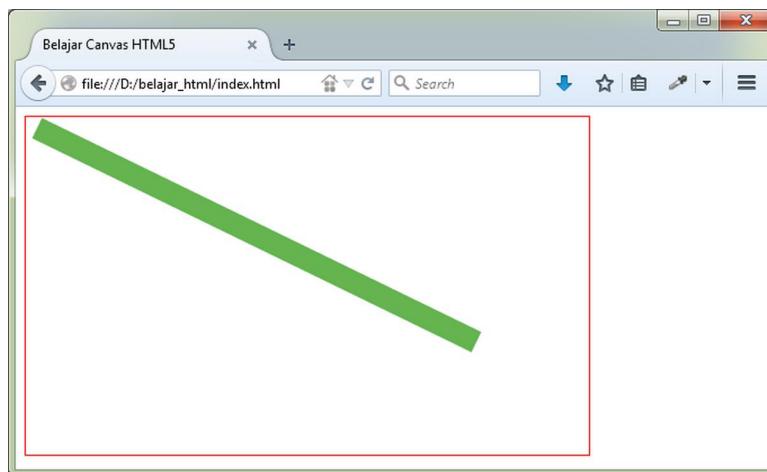
Oleh karena itu, mari kita tambahkan beberapa kode lagi untuk membuat garis tadi menjadi ‘mahakarya’ dengan mengubah tebal dan warnanya. Berikut perubahan kode JavaScript yang dibutuhkan:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Canvas HTML5</title>
6   <style>
7     canvas {
8       border: 1px solid red;
9     }
10  </style>
11 </head>
12 <body>
13  <canvas id="kertas_gambar" width="500" height="300"></canvas>
14  <script>
15    var canvasku = document.getElementById("kertas_gambar");
16    var context = canvasku.getContext("2d");
17
18    context.lineWidth = 20;
19    context.strokeStyle = "rgb(100, 180, 80)";
20
21    context.moveTo(10,10);
22    context.lineTo(400,200);
23    context.stroke();
24 </script>
```

```
25 </body>
26 </html>
```

Perintah `context.lineWidth=20` akan membuat tebal garis sebesar 20 pixel, sedangkan `context.strokeStyle="rgb(100, 180, 80)"` akan membuat garis menjadi warna hijau dengan kode RGB (100, 180, 80). Khusus untuk `context.strokeStyle`, kita juga bisa menulis kata warna dalam bahasa inggris, seperti `context.strokeStyle="red"`.



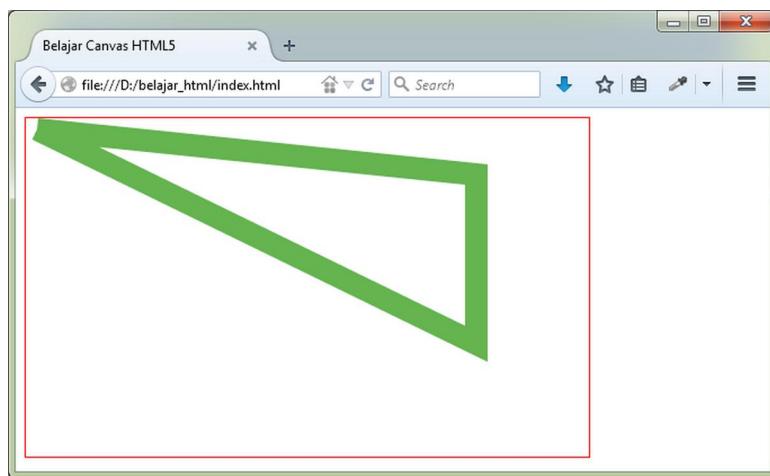
Gambar: Canvas dengan garis hijau tebal 20 pixel

Baik, 1 garis selesai, bagaimana dengan baris selanjutnya? Cukup dengan menambahkan beberapa perintah `context.lineTo()`, seperti berikut ini:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Canvas HTML5</title>
6   <style>
7     canvas {
8       border: 1px solid red;
9     }
10  </style>
11 </head>
12 <body>
13  <canvas id="kertas_gambar" width="500" height="300"></canvas>
14  <script>
15    var canvasku = document.getElementById("kertas_gambar");
16    var context = canvasku.getContext("2d");
17
18    context.lineWidth = 20;
19    context.strokeStyle = "rgb(100, 180, 80);
```

```
20      context.moveTo(10,10);
21      context.lineTo(400,200);
22      context.lineTo(400,50);
23      context.lineTo(10,10);
24      context.stroke();
25  </script>
26 </body>
27 </html>
```



Gambar: Canvas dengan ‘segitiga’ hijau tebal 20 pixel

Kali ini saya ‘mencoret’ canvas sebanyak 3 kali, yang menghasilkan gambar segitiga sembarang. Perhatikan bahwa nilai dari `context.lineTo()` adalah titik koordinat x,y.

Titik 0,0 dimulai dari sudut kiri atas, dan titik 500,300 berada di sudut kanan bawah.

14.4 Mengisi Warna Canvas

Setelah sukses membuat garis, selanjutnya kita akan mencoba ‘mengisi’ kotak segitiga yang telah kita buat pada kode sebelumnya.

Berikut adalah tambahan kode yang dibutuhkan:

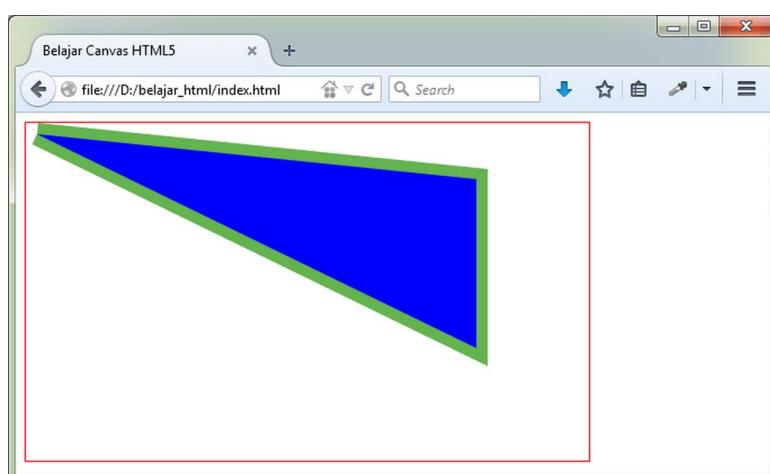
index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Canvas HTML5</title>
6   <style>
7     canvas {
8       border: 1px solid red;
```

```
9      }
10     </style>
11 </head>
12 <body>
13   <canvas id="kertas_gambar" width="500" height="300"></canvas>
14   <script>
15     var canvasku = document.getElementById("kertas_gambar");
16     var context = canvasku.getContext("2d");
17
18     context.lineWidth = 20;
19     context.strokeStyle = "rgb(100, 180, 80)";
20
21     context.moveTo(10,10);
22     context.lineTo(400,200);
23     context.lineTo(400,50);
24     context.lineTo(10,10);
25     context.stroke();
26
27     context.closePath();
28     context.fillStyle = "blue";
29     context.fill();
30   </script>
31 </body>
32 </html>
```

Perintah `context.closePath()` digunakan untuk menutup garis yang telah kita buat. Perintah `context.fillStyle="blue"` digunakan untuk memilih warna apa yang akan digunakan untuk mengisi objek yang telah di ‘close’, dalam contoh ini saya memilih warna biru (blue). Terakhir, perintah `context.fill()` digunakan untuk mulai proses ‘mengisi’ segitiga yang baru saja dibuat.

Dan, berikut adalah tampilan canvas yang dihasilkan:



Gambar: Canvas dengan ‘segitiga’ hijau dengan ‘isi’ biru

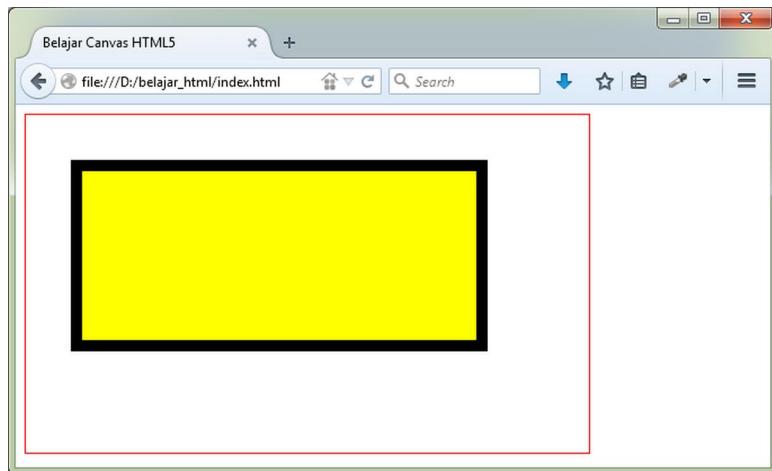
Dengan beberapa perintah JavaScript yang telah kita pelajari sejauh ini, kita sudah bisa membuat sebuah objek di dalam canvas dan ‘mewarnainya’.

Sebagai latihan, bagaimana dengan sebuah gambar persegi dengan garis tepi hitam dan berwarna kuning?

Berikut kodenya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Canvas HTML5</title>
6   <style>
7     canvas {
8       border: 1px solid red;
9     }
10  </style>
11 </head>
12 <body>
13  <canvas id="kertas_gambar" width="500" height="300"></canvas>
14  <script>
15    var canvasku = document.getElementById("kertas_gambar");
16    var context = canvasku.getContext("2d");
17
18    context.lineWidth = 20;
19    context.strokeStyle = "black";
20
21    context.moveTo(50,50);
22    context.lineTo(50,200);
23    context.lineTo(400,200);
24    context.lineTo(400,50);
25    context.lineTo(40,50);
26    context.stroke();
27
28    context.closePath();
29    context.fillStyle = "yellow";
30    context.fill();
31  </script>
32 </body>
33 </html>
```



Gambar: Canvas dengan persegi berwarna kuning dan garis tepi hitam

Jika anda perhatikan, koordinat terakhir garis adalah `context.lineTo(40,50)`, bukan `context.lineTo(50,50)`, sebagaimana koordinat awal garis. Ini karena garis itu sendiri juga memiliki lebar, sehingga kita harus mengurangkannya **10 pixel** agar kotak persegi tertutup sempurna.

14.5 Shape untuk Canvas

Shape adalah sebutan khusus untuk ‘bentuk’ seperti persegi panjang, lingkaran, atau bentuk poligon lain.

Selain menggunakan beberapa garis secara manual seperti contoh sebelum ini, kita juga bisa menggunakan perintah `context.rect()` untuk membuat persegi panjang. Berikut contohnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Canvas HTML5</title>
6   <style>
7     canvas {
8       border: 1px solid red;
9     }
10  </style>
11 </head>
12 <body>
13   <canvas id="kertas_gambar" width="500" height="300"></canvas>
14   <script>
15     var canvasku = document.getElementById("kertas_gambar");
16     var context = canvasku.getContext("2d");
17   
```

```
18     context.lineWidth = 20;
19     context.strokeStyle = "black";
20
21     context.rect(50, 50, 350, 150);
22     context.stroke();
23
24     context.fillStyle = "yellow";
25     context.fill();
26 </script>
27 </body>
28 </html>
```

Jika anda menjalankan kode di atas, hasilnya sama persis dengan persegi panjang yang dibuat sebelum ini, namun kali ini kita menggunakan perintah `context.rect()`.

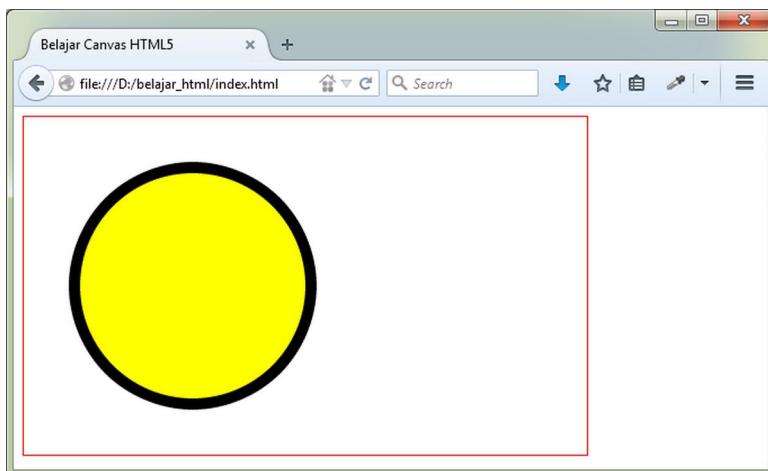
Perintah `context.rect(50, 50, 350, 150)` berarti ‘*buat persegi panjang dengan koordinat awal di 50,50 dengan lebar 350 pixel, dan tinggi 150 pixel*’.

Sedangkan untuk membuat lingkaran, kita bisa menggunakan perintah `context.arc()` seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Canvas HTML5</title>
6   <style>
7     canvas {
8       border: 1px solid red;
9     }
10  </style>
11 </head>
12 <body>
13  <canvas id="kertas_gambar" width="500" height="300"></canvas>
14  <script>
15    var canvasku = document.getElementById("kertas_gambar");
16    var context = canvasku.getContext("2d");
17
18    context.lineWidth = 20;
19    context.strokeStyle = "black";
20
21
22    context.arc(150,150,100,0,2*Math.PI);
23    context.stroke();
24
```

```
25     context.fillStyle = "yellow";
26     context.fill();
27 </script>
28 </body>
29 </html>
```



Gambar: Canvas dengan lingkaran berwarna kuning dan garis tepi hitam

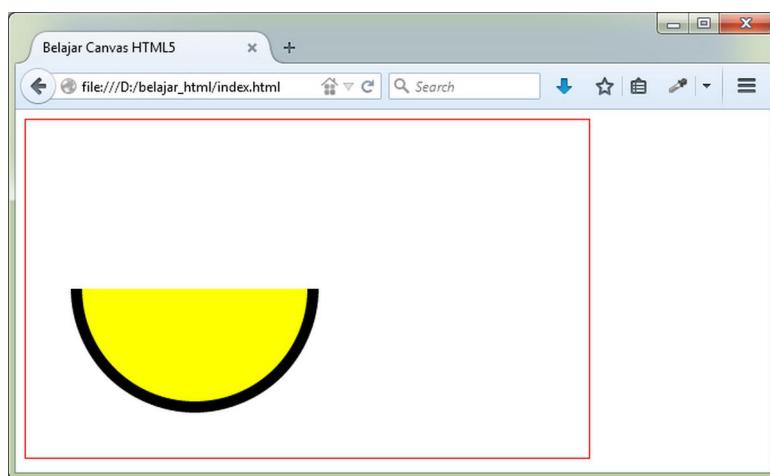
Perintah `context.arc(150,150,100,0,2*Math.PI)` artinya: '*buat lingkaran dengan titik pusat di 150,150 dengan diameter 100 pixel, sudut awal 0 dan sudut akhir 2*Math.PI*'. Khusus untuk perhitungan sudut ini menggunakan satuan **radian**.

Bagaimana dengan setengah lingkaran? Kita tinggal mengubah posisi sudut akhir menjadi **Math.PI**, seperti contoh berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Canvas HTML5</title>
6   <style>
7     canvas {
8       border: 1px solid red;
9     }
10  </style>
11 </head>
12 <body>
13   <canvas id="kertas_gambar" width="500" height="300"></canvas>
14   <script>
15     var canvasku = document.getElementById("kertas_gambar");
16     var context = canvasku.getContext("2d");
17
18     context.lineWidth = 20;
```

```
19     context.strokeStyle = "black";
20
21
22     context.arc(150,150,100,0,Math.PI);
23     context.stroke();
24
25     context.fillStyle = "yellow";
26     context.fill();
27 </script>
28 </body>
29 </html>
```



Gambar: Canvas dengan setengah lingkaran berwarna kuning dan garis tepi hitam

14.6 Membuat Teks di dalam Canvas

Tidak hanya garis dan shape, kita juga bisa menambahkan teks ke dalam canvas menggunakan perintah `context.font()` dan `context.fillText()`.

Berikut contohnya:

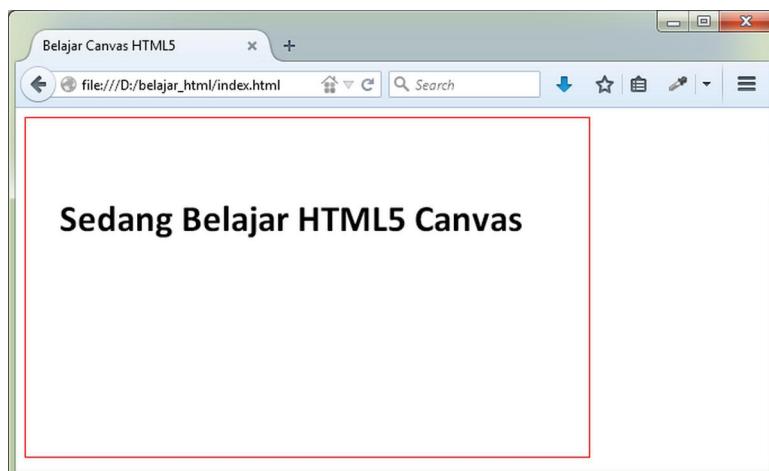
index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Canvas HTML5</title>
6   <style>
7     canvas {
8       border: 1px solid red;
9     }
10  </style>
```

```

11 </head>
12 <body>
13   <canvas id="kertas_gambar" width="500" height="300"></canvas>
14   <script>
15     var canvasku = document.getElementById("kertas_gambar");
16     var context = canvasku.getContext("2d");
17
18     context.font = 'bold 25pt Calibri';
19     context.fillText('Sedang Belajar HTML5 Canvas', 30, 100);
20   </script>
21 </body>
22 </html>

```



Gambar: Canvas dengan font tulisan

Perintah `context.font="bold 25pt Calibri"` artinya: “*pilih jenis font calibri sebesar 25point dalam bentuk tebal (bold)*”.

Sedangkan perintah `context.fillText('Sedang Belajar HTML5 Canvas', 30, 100)` berarti: “*buat teks ‘Sedang Belajar HTML5 Canvas’, dimulai dari koordinat 30,100*”

14.7 Penutup: HTML5 Canvas

Perintah JavaScript untuk canvas yang kita pelajari disini hanyalah sebagian kecil dari fungsi-fungsi yang ada di dalam HTML5. Namun saya rasa cukup untuk membekali anda tentang cara penggunaan **canvas**.

Saya sengaja tidak membahas lebih jauh karena kode-kode yang digunakan tidak terlalu cocok dibahas dalam buku pengantar HTML. Jika anda ingin mempelajari HTML5 canvas lebih jauh, bisa mengunjungi www.html5canvastutorials.com¹.

Beberapa fitur canvas yang belum dibahas seperti membuat *gradient*, *kurva* (*quadratic curve* dan *bezier curve*), *transformations* (*scale*, *rotate*, *mirror*), *shadow*, serta *animation*. Tapi sebaiknya

¹<http://www.html5canvastutorials.com>

anda sudah memiliki bekal JavaScript agar dapat menguasai HTML5 canvas dengan lebih sempurna.

Beberapa contoh penerapan HTML5 yang menarik bisa anda lihat di sini code.tutsplus.com²

Untuk membuat gambar yang lebih rumit menggunakan canvas, juga banyak tersedia tool atau library, seperti [Fabric.js](http://fabricjs.com)³ dan [KineticJS](http://kineticjs.com)⁴.

Dalam bab ini kita telah membahas cara penggunaan fitur canvas di dalam HTML. Selain menggunakan canvas, tersedia juga format gambar SVG, yang penulisannya mirip seperti tag-tag HTML. Inilah yang akan kita bahas dalam bab selanjutnya.

²<http://code.tutsplus.com/articles/21-ridiculously-impressive-html5-canvas-experiments--net-14210>

³<http://fabricjs.com>

⁴<http://kineticjs.com>

15. Scalable Vector Graphics (SVG)

Selain menggunakan **HTML5 Canvas**, kita juga bisa ‘menggambar’ menggunakan **SVG (Scalable Vector Graphics)**. Dalam bab ini saya akan membahas apa itu **SVG** dan bagaimana cara penggunaannya.

15.1 Pengertian SVG

Scalable Vector Graphics atau sering disebut dengan singkatan **SVG**, adalah sebuah teknologi yang memungkinkan kita membuat gambar grafis menggunakan tag-tag **XML**. Sebenarnya **SVG** bukanlah bagian dari **HTML**. Mirip seperti **CSS** dan **JavaScript**, **SVG** merupakan bagian dari teknologi web development.

SVG sudah hadir sebelum **HTML5**, tepatnya sejak tahun 1999 yang dikembangkan oleh **W3C** (Organisasi yang juga mengembangkan standar **HTML** dan **CSS**). Sama seperti **HTML**, **SVG** juga ditulis menggunakan tag-tag seperti `<svg>`, `<line>`, `<circle>`, dst. Ini karena **SVG** menggunakan format penulisan **XML** yang jika ditelusuri merupakan saudara jauh **HTML** (silahkan anda membaca kembali tentang sejarah **HTML** di bab 2).

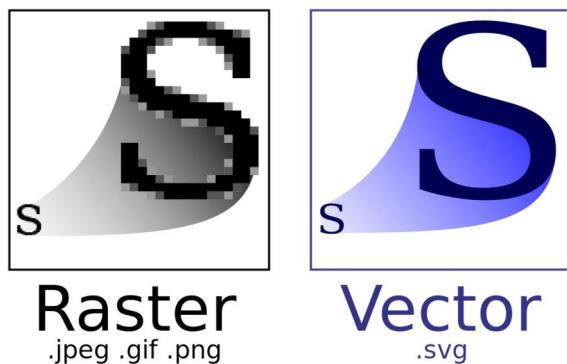
Walaupun bukan bagian dari **HTML**, saya merasa **SVG** ini cukup menarik untuk dibahas. Penggunaan **SVG** tidak hanya untuk web programming saja. Designer grafis yang sering menggunakan aplikasi seperti **Adobe Illustrator** atau **Inkscape** sudah kenal dekat dengan format gambar ini.

15.2 Raster Image vs Vector Image

Salah satu keunggulan dari **SVG** adalah: gambar tidak akan pecah ketika di zoom. Ini karena cara kerja **SVG** berbeda dengan format gambar yang sering kita gunakan, seperti **gif**, **jpg** atau **png**.

Gambar yang disimpan dalam format **jpg** (termasuk juga gambar hasil **HTML5 Canvas**) termasuk ke dalam kelompok **Raster Image** atau **Bitmap**. Dalam format ini, gambar disimpan dalam bentuk pixel-pixel kecil. Ketika dilakukan pembesaran, pixel-pixel ini ‘dipaksa’ membesar sehingga gambar menjadi kurang jelas.

SVG merupakan format gambar yang masuk ke dalam kelompok **Vector Image**. Disini gambar disimpan menggunakan titik-titik koordinat atau persamaan matematis. Komputer memproses koordinat ini dan menampilkannya menjadi sebuah gambar. Ketika di zoom, komputer ‘paham’ bahwa ini adalah garis dari titik A ke titik B, jadi gambar tinggal disesuaikan dan akan tetap tajam.



Gambar: Perbedaan Raster Image dengan Vector Image, sumber: wikipedia

Dibalik keunggulannya, SVG juga memiliki kelemahan. Kita tidak bisa menyimpan gambar foto pemandangan ke dalam SVG, karena ini terlalu kompleks untuk dibuat persamaan matematikanya. Oleh karena itulah gambar-gambar SVG yang anda jumpai biasanya hanya berupa icon, teks, sketsa atau gambar 2 dimensi. Gambar dunia nyata seperti foto tetap harus disimpan ke dalam format **raster image** seperti *jpg*.



Dalam bab ini kita akan belajar cara membuat gambar SVG dari teks editor. Untuk gambar yang kompleks, anda bisa menggunakan aplikasi pengolah gambar khusus vector, seperti [Adobe Illustrator¹](#) atau [Inkscape²](#).

15.3 SVG Element

Cukup tentang teori seputar SVG, mari kita masuk ke praktek. Jika dalam **HTML5 Canvas** kita menggunakan tag `<canvas>`, maka untuk **SVG** kita menggunakan (tidak salah lagi..) tag `<svg>`.

Berikut contoh halaman HTML dengan `<svg>` element:

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8" />
5   <title>Belajar HTML</title>
6 </head>
7 <body>
8   <svg xmlns="http://www.w3.org/2000/svg" >
9
10  </svg>
11 </body>
12 </html>

```

Selain tag pembuka `<svg>` dan penutup `</svg>`, terdapat sebuah atribut **xmlns**. Atribut ini dikenal dengan istilah: [namespace declarations³](#).

¹<http://www.adobe.com/sea/products/illustrator.html>

²<https://inkscape.org/en/>

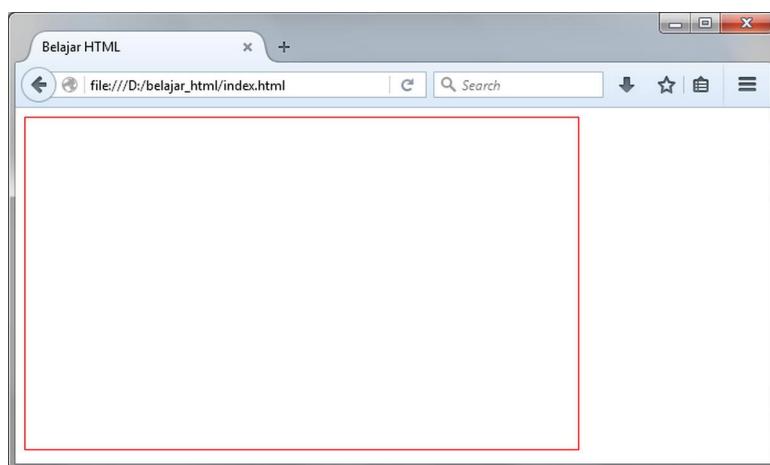
³https://developer.mozilla.org/en/docs/Web/SVG/Namespace_Declarations

SVG menggunakan ‘aturan’ dokumen XML, oleh karena itu SVG harus mengikuti syarat-syarat sebuah element XML. **Namespace declarations** berfungsi untuk menginformasikan web browser bahwa seluruh element yang berada diantara tag <svg> dan </svg> merupakan element XML yang di definisikan oleh <http://www.w3.org/2000/svg>.

Sebenarnya, tanpa menulis atribut **xmlns** inipun mayoritas web browser tetap bisa memproses SVG. Ini karena element <svg> kita tulis di dalam dokumen HTML. Jika SVG dibuat ke dalam file terpisah, atribut ini harus ditulis (kita akan pelajari nanti).

Jika anda menjalankan kode di atas, belum tampak apa-apa di web browser. Ini karena kita harus mendefinisikan tinggi dan lebar ‘kertas gambar’ <svg> menggunakan atribut **width** dan **height**. Berikut perubahannya:

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8" />
5      <title>Belajar HTML</title>
6      <style>
7          svg {
8              border: 1px solid red;
9          }
10     </style>
11 </head>
12 <body>
13     <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
14
15     </svg>
16 </body>
17 </html>
```



Gambar: ‘Kertas Gambar’ untuk SVG

Saya melakukan hal yang sama seperti pada bab **Canvas**. Atribut **width="500"** dan **height="300"** pada tag <svg> digunakan untuk menyiapkan ‘kertas gambar’ dengan lebar 500 pixel dan tinggi

300 pixel. Saya juga menggunakan kode CSS untuk membuat bingkai luar dari kertas gambar ini (border merah).

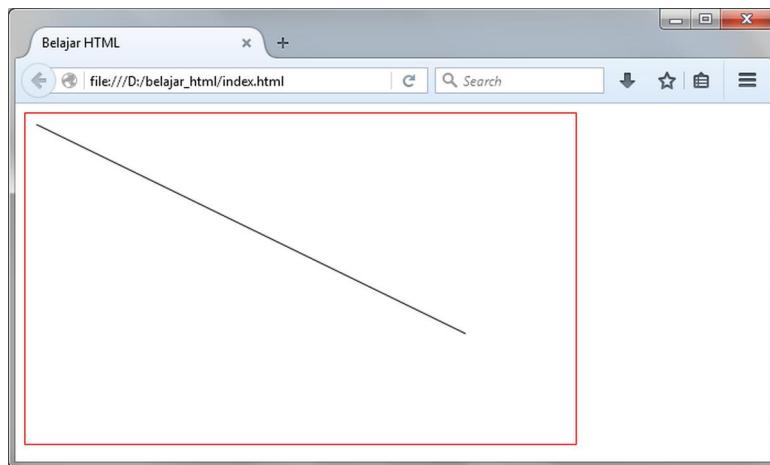
Saatnya menggambar!

15.4 Membuat Garis dengan SVG

Membuat gambar/grafik dengan SVG jauh lebih simple jika dibandingkan dengan <canvas>. Kita tidak perlu menggunakan JavaScript. Walaupun sebenarnya SVG-pun bisa diprogram melalui JavaScript.

Berikut contoh kode SVG untuk membuat sebuah garis:

```
1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <line x1="10" y1="10" x2="400" y2="200" stroke="black" />
3 </svg>
```



Gambar: Membuat garis dengan tag <line> SVG

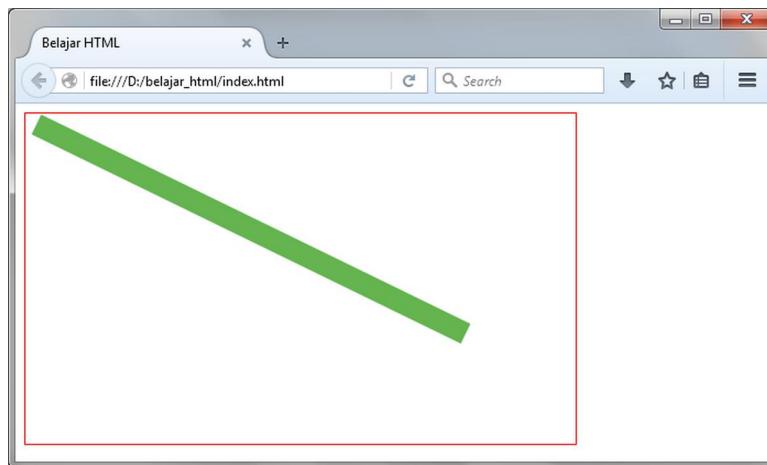
Untuk membuat garis, kita menggunakan tag <line>. Tag ini memiliki beberapa atribut:

- **x1** dan **y1**: menentukan koordinat awal garis.
- **x2** dan **y2**: menentukan koordinat akhir garis.
- **stroke**: menentukan warna garis.

Kode di atas bisa dibaca: *buat garis hitam dari titik 10,10 hingga 400,200*.

Untuk menebalkan garis, bisa ditambahkan atribut **stroke-width**. Nilai atribut ini berupa angka dalam satuan pixel:

```
1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <line x1="10" y1="10" x2="400" y2="200"
3     stroke="rgb(100, 180, 80)" stroke-width="20"/>
4 </svg>
```

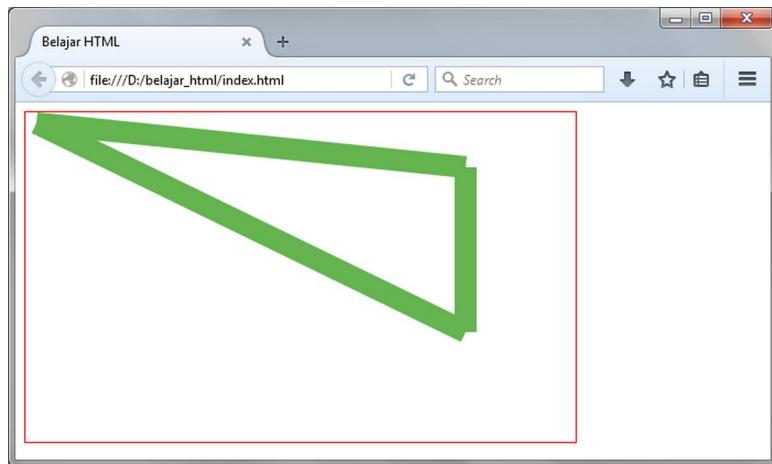


Gambar: Garis tampil tebal dengan atribut stroke-width

Selain menulis `stroke-width="20"` yang akan membuat tebal garis menjadi 20 pixel, saya juga mengubah warna menggunakan atribut `stroke="rgb(100, 180, 80)"`. Disini saya menggunakan aturan penulisan warna RGB yang biasa digunakan pada CSS.

Untuk membuat lebih dari 1 garis, kita tinggal menulis tag `<line>` beberapa kali. Tentunya dengan titik koordinat yang berbeda-beda:

```
1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <line x1="10" y1="10" x2="400" y2="200"
3     stroke="rgb(100, 180, 80)" stroke-width="20" />
4   <line x1="400" y1="200" x2="400" y2="50"
5     stroke="rgb(100, 180, 80)" stroke-width="20" />
6   <line x1="400" y1="50" x2="10" y2="50"
7     stroke="rgb(100, 180, 80)" stroke-width="20" />
8 </svg>
```

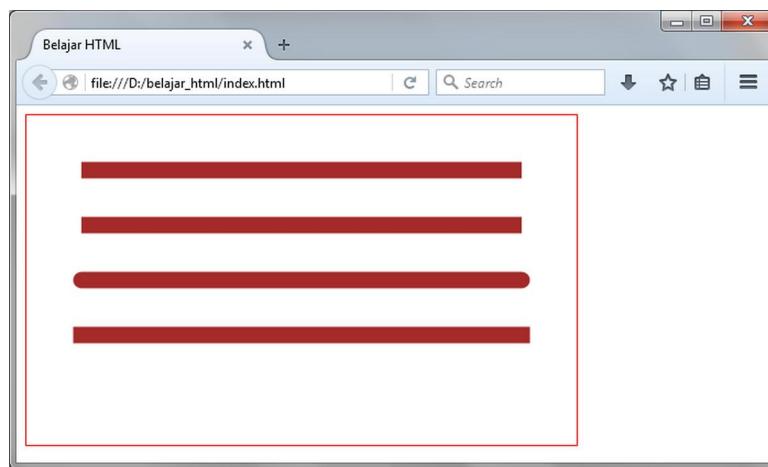


Gambar: 3 garis tag <line> yang membentuk segitiga

Disini saya mencoba meniru tampilan yang kita buat pada bab **Canvas**, namun terdapat sedikit perbedaan. Ujung setiap garis tampak tidak menyatu, karena masing-masingnya memang merupakan garis terpisah. Jika kita ingin agar garis ini berlanjut satu sama lain (membentuk segitiga), SVG menyediakan tag <polygon> yang nantinya juga akan kita pelajari.

Atribut selanjutnya yang bisa ditambahkan adalah: **stroke-linecap**. Atribut ini berfungsi untuk mengatur bentuk ujung garis. Terdapat 3 nilai yang bisa digunakan, yakni: **butt** (pilihan default), **round** dan **square**. Berikut contoh penggunaannya:

```
1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <line x1="50" y1="50" x2="450" y2="50"
3     stroke="brown" stroke-width="15" />
4   <line x1="50" y1="100" x2="450" y2="100"
5     stroke="brown" stroke-width="15" stroke-linecap="butt"/>
6   <line x1="50" y1="150" x2="450" y2="150"
7     stroke="brown" stroke-width="15" stroke-linecap="round"/>
8   <line x1="50" y1="200" x2="450" y2="200"
9     stroke="brown" stroke-width="15" stroke-linecap="square"/>
10 </svg>
```



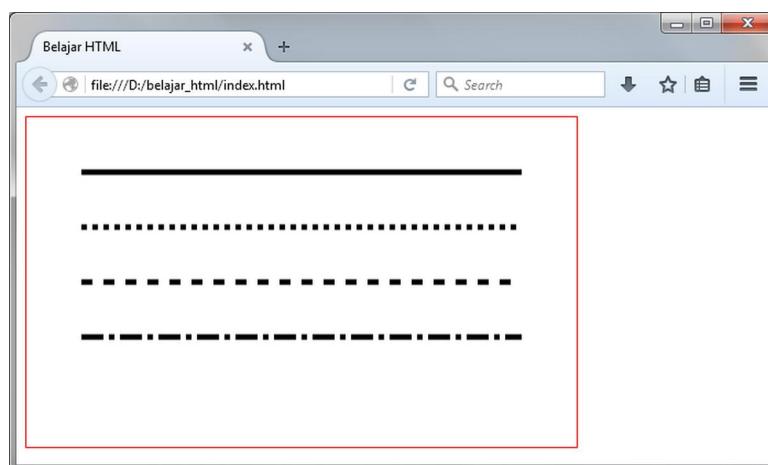
Gambar: Efek penambahan atribut stroke-linecap

Seperti yang terlihat, jika atribut **stroke-linecap** tidak ditulis, SVG akan menggunakan **stroke-linecap="butt"**. Untuk **stroke-linecap="round"**, ujung garis terlihat membulat. Sedangkan untuk **stroke-linecap="square"**, ujung garis berupa kotak persegi.

Atribut terakhir yang bisa kita gunakan untuk garis adalah **stroke-dasharray**. Atribut ini berfungsi untuk membuat garis putus-putus. Berikut contohnya:

```

1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <line x1="50" y1="50" x2="450" y2="50"
3     stroke="black" stroke-width="5" />
4   <line x1="50" y1="100" x2="450" y2="100"
5     stroke="black" stroke-width="5" stroke-dasharray="5,5"/>
6   <line x1="50" y1="150" x2="450" y2="150"
7     stroke="black" stroke-width="5" stroke-dasharray="10,10"/>
8   <line x1="50" y1="200" x2="450" y2="200"
9     stroke="black" stroke-width="5" stroke-dasharray="20,5,5,5"/>
10 </svg>
```



Gambar: Membuat efek garis putus-putus menggunakan atribut stroke-dasharray

Atribut `stroke-dasharray="5,5"` bisa dibaca: *buat garis sepanjang 5 pixel, kemudian kosong sepanjang 5 pixel, begitu seterusnya hingga titik koordinat berakhir*. Begitu juga dengan atribut `stroke-dasharray="10,10"` yang artinya *buat garis berselang seling garis putus-putus sepanjang 10 pixel*.

Pada garis ketiga, saya menggunakan `stroke-dasharray="20,5,5,5"`. Ini artinya *garis tampil sepanjang 20 pixel, kemudian kosong 5 pixel, garis kembali 5 pixel, kosong 5 pixel, dan berulang hingga selesai*. Hasilnya cukup unik, anda bisa mencoba kombinasi untuk membuat efek-efek menarik lainnya.

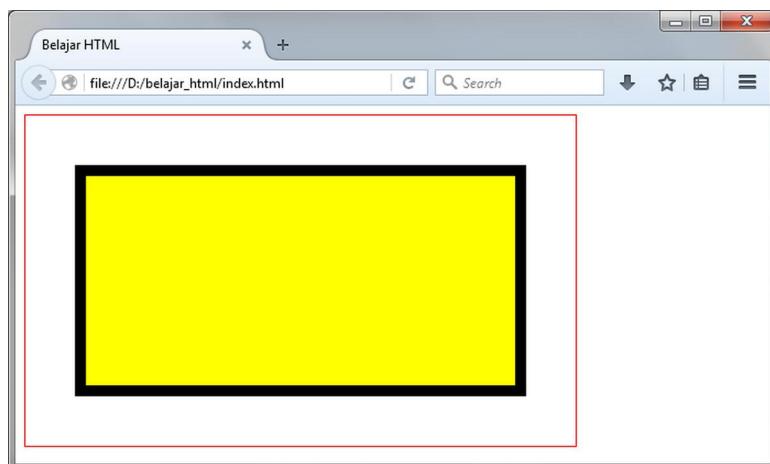
15.5 Membuat Persegi dengan SVG

Untuk membuat kotak persegi, SVG menyediakan tag `<rect>`, yakni singkatan dari **rectangular**. Berikut contoh penggunaannya:

```

1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <rect x="50" y="50" width="400" height="200"
3     stroke="black" stroke-width="10" fill="yellow"/>
4 </svg>

```



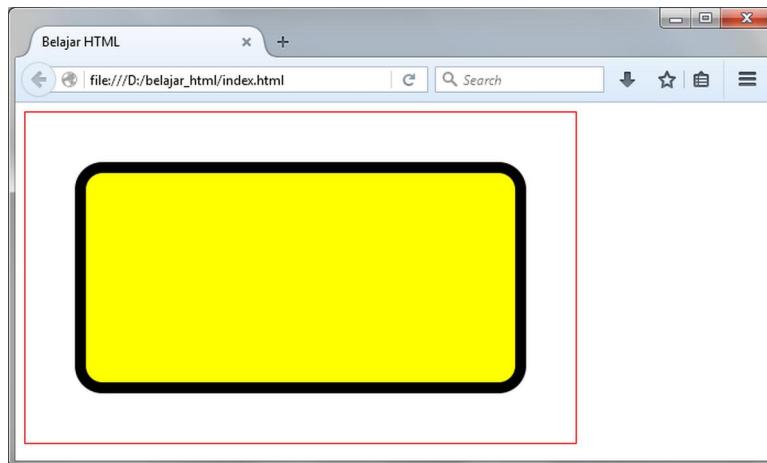
Gambar: Kotak persegi yang dibuat dari tag `<rect>` SVG

Berikut penjelasan dari atribut yang digunakan:

- `x` dan `y`: koordinat titik awal persegi (titik kanan atas).
- `width`: lebar persegi.
- `height`: tinggi persegi.
- `stroke`: warna garis tepi (border).
- `stroke-width`: lebar garis tepi.
- `fill`: warna dari isi persegi.

Tag `<rect>` juga bisa ditambahkan atribut `rx` dan `ry`. Fungsinya, untuk membuat sudut melengkung (*rounding corner*). Nilai kedua atribut ini dalam satuan pixel, berupa seberapa besar jari-jari kelengkungan yang ingin dibuat. Berikut contohnya:

```
1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <rect x="50" y="50" width="400" height="200" rx="20" ry="20"
3     stroke="black" stroke-width="10" fill="yellow"/>
4 </svg>
```



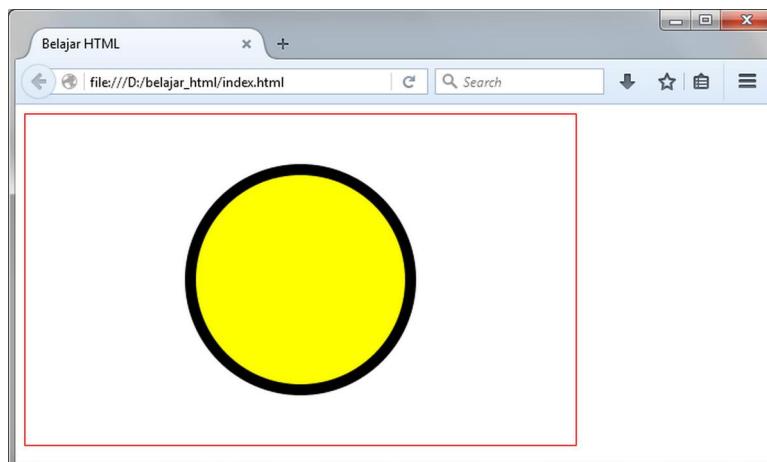
Gambar: Tambahan atribut rx dan ry untuk membuat rounding corner

Kali ini ujung persegi terlihat membulat.

15.6 Membuat Lingkaran dan Elips dengan SVG

Untuk membuat gambar lingkaran, kita bisa menggunakan tag `<circle>`. Berikut contohnya:

```
1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <circle cx="250" cy="150" r="100"
3     stroke="black" stroke-width="10" fill="yellow"/>
4 </svg>
```



Gambar: Membuat gambar lingkaran menggunakan tag `<circle>`

Terdapat 3 atribut baru untuk tag `<circle>`, yakni:

- cx dan cy: titik koordinat awal lingkaran.
- r: radius lingkaran dalam satuan pixel.

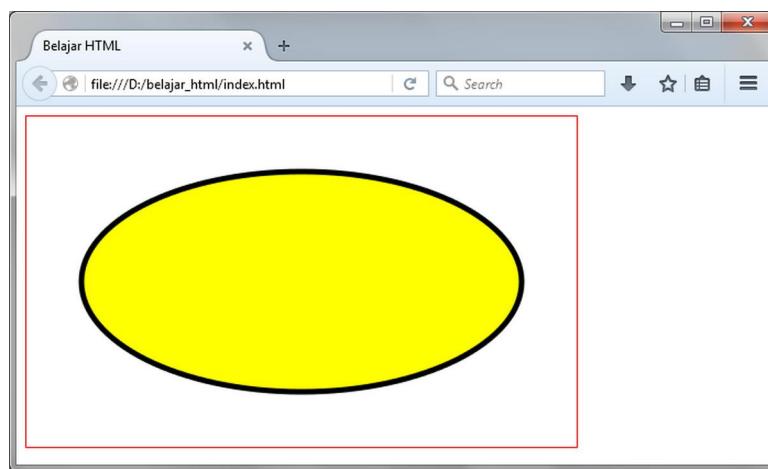
Sedangkan untuk atribut **stroke**, **stroke-width**, dan **fill** fungsinya sama seperti pada tag `<rect>`.

Bagaimana dengan elips? Kita bisa menggunakan tag `<ellipse>`, seperti berikut ini:

```

1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <ellipse cx="250" cy="150" rx="200" ry="100"
3     stroke="black" stroke-width="5" fill="yellow"/>
4 </svg>

```



Gambar: Membuat gambar elips menggunakan tag `<ellipse>`

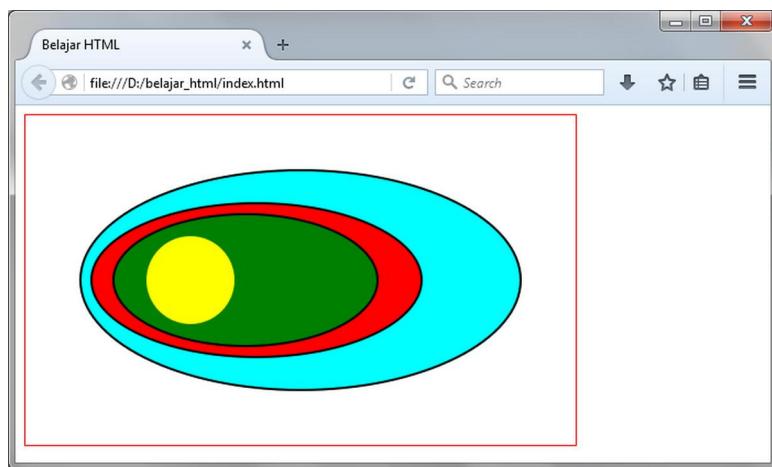
Sedikit berbeda dengan tag `<rect>`, untuk tag `<ellipse>` terdapat 2 radius, yakni radius untuk sumbu x (atribut **rx**) dan radius untuk sumbu y (atribut **ry**). Kedua radius ini harus berbeda agar terbentuk elips. Jika keduanya memiliki nilai sama, hasilnya berupa sebuah lingkaran.

Dengan mengkombinasikan tag `<rect>` dan `<ellipse>` serta sedikit kreatifitas, kita bisa membuat efek menarik:

```

1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <ellipse cx="250" cy="150" rx="200" ry="100"
3     stroke="black" stroke-width="2" fill="aqua"/>
4   <ellipse cx="210" cy="150" rx="150" ry="70"
5     stroke="black" stroke-width="2" fill="red"/>
6   <ellipse cx="200" cy="150" rx="120" ry="60"
7     stroke="black" stroke-width="2" fill="green"/>
8   <circle cx="150" cy="150" r="40"
9     fill="yellow"/>
10 </svg>

```



Gambar: 1 lingkaran dan 3 elips menggunakan SVG

Disini saya ‘menumpuk’ 1 lingkaran dengan 3 elips. Perhatikan bahwa tag-tag SVG diproses dari atas hingga bawah, oleh karena itu saya harus menulis tag `<circle>` pada urutan terakhir, agar lingkaran ini tampil di atas ketiga elips.

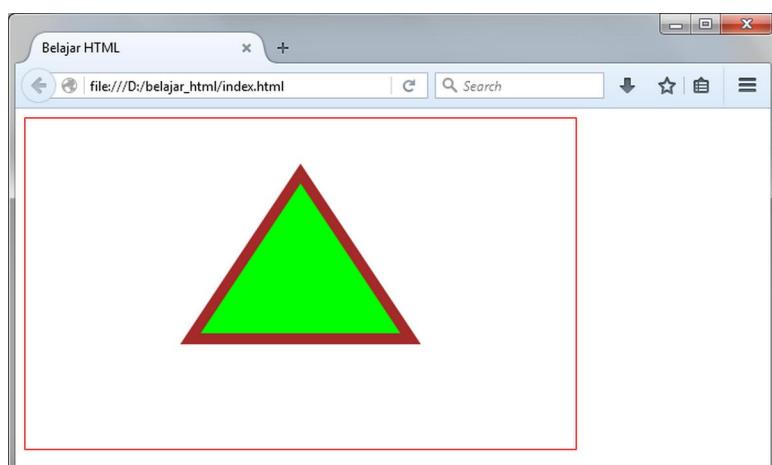
15.7 Membuat Polygon dengan SVG

Polygon adalah istilah dalam SVG untuk membuat gambar dengan berbagai titik koordinat. Ini dibuat menggunakan tag `<polygon>`. Sebagai contoh, segitiga bisa disebut sebagai poligon, karena memiliki 3 titik. Berikut contoh penggunaannya:

```

1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <polygon points="250,50 350,200 150,200"
3     stroke="brown" stroke-width="10" fill="lime"/>
4 </svg>

```



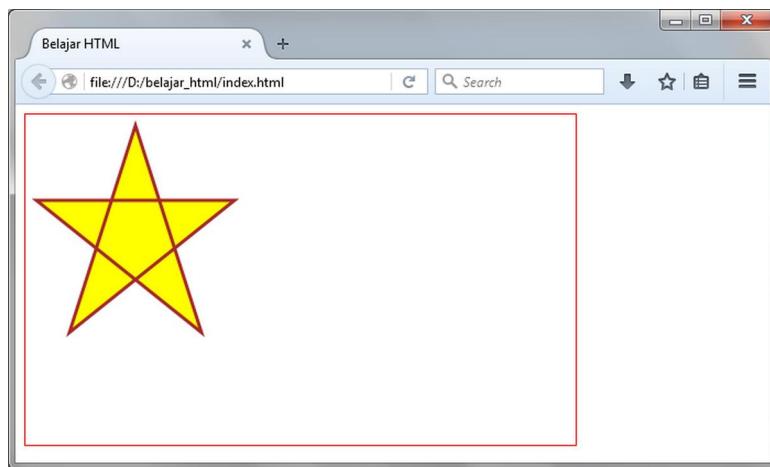
Gambar: Membuat gambar segitiga menggunakan tag `<polygon>`

Atribut **points** di dalam tag `<polygon>` berfungsi untuk membuat titik koordinat. Titik koordinat ini bergantian dari titik $x_1,y_1,x_2, y_2, x_3, y_3$, dst. Atribut `points="250,50 350,200`

150,200" artinya, saya membuat poligon dengan 3 titik, titik pertama pada koordinat 250,50, titik kedua pada 350,200 dan titik ketiga pada 150,200. Saat titik ketiga dibuat, poligon otomatis menghubungkannya kembali dengan titik pertama.

SVG tidak membatasi berapa jumlah titik koordinat yang dibuat. Kita bisa membuat gambar yang cukup kompleks seperti berikut ini:

```
1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <polygon points="100,10 40,198 190,78 10,78 160,198"
3     stroke="brown" stroke-width="3" fill="yellow"/>
4 </svg>
```



Gambar: Membuat gambar bintang dengan tag <polygon> SVG

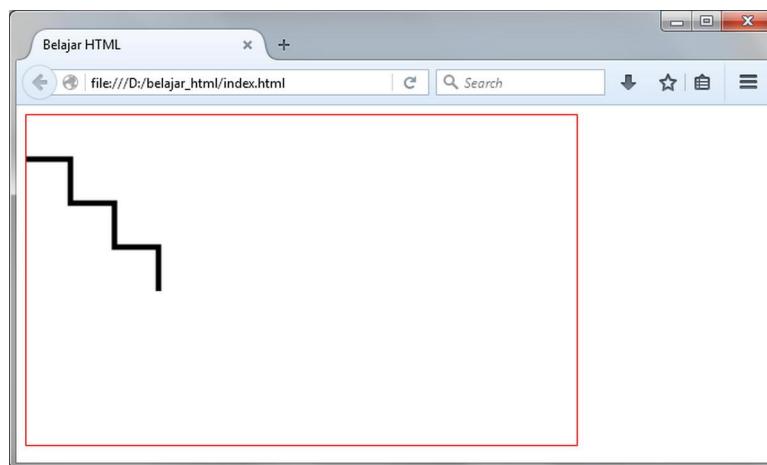
Untuk menghasilkan gambar bintang di atas, saya menggunakan 5 titik koordinat. Diperlukan perhitungan yang pas agar kelima titik ini simetris dan berbentuk gambar bintang.

15.8 Membuat Garis Bersambung dengan SVG

Ketika mempelajari tag <line>, saya mencoba membuat 3 garis yang tampak menyatu (menjadi segitiga). Akan tetapi hasilnya berupa 3 garis yang berbeda, bukan satu kesatuan.

Untuk membuat sebuah garis yang saling bersambung, SVG menyediakan tag <polyline>. Berikut contoh penggunaannya:

```
1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <polyline points="0,40 40,40 40,80 80,80 80,120 120,120 120,160"
3     stroke="black" stroke-width="5" fill="white"/>
4 </svg>
```



Gambar: Membuat garis bersambung dengan tag `<polyline>` SVG

Mirip seperti tag `<polygon>`, dalam tag `<polyline>` juga menggunakan atribut **points**. Nilai dari atribut ini berupa pasangan koordinat x dan y. Atribut **points** dalam kode di atas bisa dibaca: *Mulai dari titik 0,40, buat garis lurus hingga 40,40, kemudian lanjut ke titik 40,80, dst.*

15.9 Membuat Teks dengan SVG

Selain menampilkan gambar, kita juga bisa menampilkan teks menggunakan tag `<text>`. Berikut contohnya:

```
1 <svg xmlns="http://www.w3.org/2000/svg" width="500" height="300" >
2   <text x="50" y="150" fill="blue" font-family="Copperplate Gothic Bold"
3     font-size="55">Belajar SVG</text>
4   <text x="300" y="290" fill="black" font-family="Cambria"
5     font-size="40">DuniaIlkom</text>
6 </svg>
```



Gambar: Membuat teks tulisan dengan tag `<text>` SVG

Disini saya menulis 2 buat tag `<text>` dengan teks “Belajar SVG” dan “DuniaIlkom”. Di dalam tag `<text>` ini saya menggunakan 2 buah atribut baru, yakni `font-family` yang berfungsi untuk menentukan jenis font, serta atribut `font-size` untuk mengatur ukuran font (dalam satuan pixel).

15.10 Menggunakan External SVG

Dalam contoh-contoh yang kita jalankan sampai sekarang, gambar SVG berada langsung di halaman HTML. Dalam prakteknya, gambar ini umumnya merupakan dalam file terpisah, mirip seperti gambar `.jpg` maupun `.png`.

Kita pun bisa memindahkan gambar SVG ke sebuah file terpisah, kemudian menginputnya menggunakan tag `` seperti gambar biasa. File gambar SVG ini disimpan dengan extension `.svg`. Silahkan anda ketik kode berikut ke dalam sebuah file baru:

bendera.svg

```
1 <svg width="350" height="250" xmlns="http://www.w3.org/2000/svg">
2   <rect fill="white" x="10" y="10" width="300" height="200"
3     stroke-width="1" stroke="black" />
4   <rect fill="red" x="10" y="10" width="300" height="100" />
5   <rect fill="white" x="10" y="110" width="300" height="100" />
6 </svg>
```

Sebelum menjalankan file ini, dapatkah anda menebak gambar apa yang saya buat?

Dari kode di atas terdapat 3 buah tag `<rect>`, artinya ada 3 buah kotak persegi. Kotak pertama berupa persegi panjang dengan dimensi 300×200 pixel, isi kotak berwarna putih (`white`) dan garis tepi hitam. Titik kanan atas kotak berada di titik 10, 10.

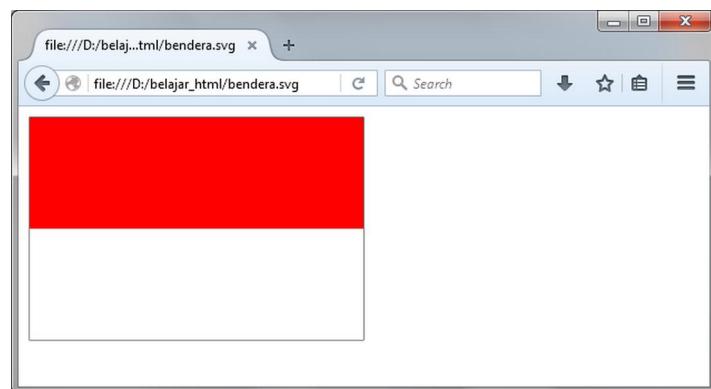
Kotak kedua berukuran 300×100 , memiliki warna merah (`red`) dan juga dimulai dari titik 10,10. Artinya, kotak ini akan menimpa kotak pertama, tapi hanya setengah bagian atas.

Kotak ketiga juga berukuran 300×10 , berwarna putih dan dimulai dari titik 10, 110. Artinya, kotak ini juga menimpa kotak pertama namun setengah bagian bawah.

Bisakah anda menebak bentuk akhir dari ketiga kotak ini? Yup, berupa kotak merah-putih yang menyerupai bendera **Indonesia**.

Save-lah kode di atas dengan nama file: `bendera.svg`. Ingat, kali ini kita tidak membuat dokumen HTML, tapi sebuah dokumen SVG.

Dokumen SVG tidak memerlukan deklarasi `<!DOCTYPE html>`, tapi langsung tag `<svg>`. Atribut `xmlns="http://www.w3.org/2000/svg"` wajib ditulis. Jika tidak, gambar SVG tidak akan tampil di web browser. Berikut tampilan file `bendera.svg` ketika di akses dari web browser:



Gambar: File bendera.svg langsung ditampilkan di web browser

Selanjutnya, buat sebuah file HTML baru, dan akses file **bendera.svg** menggunakan tag ``, seperti saat kita mengakses gambar *jpg*:

index.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8" />
5     <title>Belajar HTML</title>
6     <style>
7       svg {
8         border: 1px solid red;
9       }
10    </style>
11  </head>
12  <body>
13    <h2>Jayalah negeriku, Jayalah bangsaku...</h2>
14    
15  </body>
16 </html>
```



Gambar: Jayalah negeriku, Jayalah bangsaku... Indonesia!

Cara seperti ini umum digunakan untuk memisahkan file gambar SVG dengan kode HTML. Penggunaannya juga tidak berbeda dengan format gambar yang lainnya seperti *.jpg* maupun *.png*.

Sebagai praktik tambahan, silahkan anda zoom gambar bendera ini menggunakan fitur zoom di web browser, gambar tersebut tidak akan pecah.

Dalam bab ini kita telah melihat sekilas cara membuat gambar menggunakan **Scalable Vector Graphics**. Tentu saja ini masih sekilas dari apa yang bisa dihasilkan dari SVG.

Dalam penggunaan sehari-hari, akan lebih gampang membuat gambar SVG menggunakan aplikasi pengolah gambar vector seperti **Adobe Illustrator** atau **Inkscape**, kemudian baru diakses dari tag ``, seperti contoh terakhir kita. Namun dengan memahami teknologi di balik SVG, setidaknya anda bisa paham prinsip kerja dibalik format gambar yang ‘masih bersaudara’ dengan HTML ini.

SVG juga bisa dikombinasikan dengan JavaScript untuk membuat fitur interaktif, seperti gambar yang bisa di-klik, di-drag, membuat animasi, dll. Jika anda tertarik, bisa melihat hasilnya ke: snapsvg.io⁴ atau [bonsaijs.org](http://demos.bonsaijs.org)⁵.

⁴<http://snapsvg.io/demos/>

⁵<http://demos.bonsaijs.org>

16. HTML5 Semantic Tag (Layout)

Dalam bab ini kita akan membahas tentang cara membuat struktur halaman (layout) HTML menggunakan semantic tag HTML5.

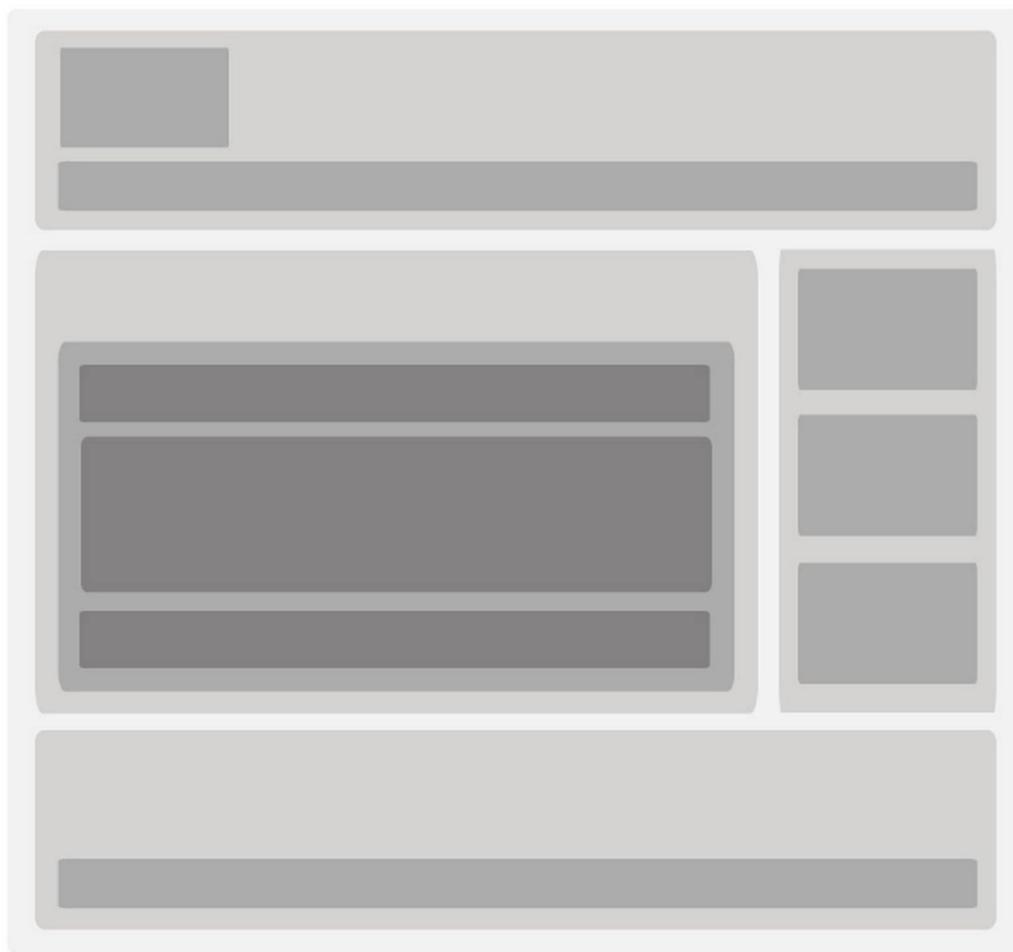
Saya sengaja menempatkan pembahasan ini di bagian akhir buku, karena apa yang akan kita pelajari kali ini juga berfungsi sebagai ‘penyatu’ dari seluruh materi yang telah dipelajari dari bab pertama. Kita akan melihat bagaimana setiap tag HTML disatukan ke dalam sebuah halaman utuh.

16.1 Mengenal Struktur Halaman Web

Seperti yang umum kita lihat, sebuah halaman website terdiri dari struktur halaman seperti *header*, *footer*, *sidebar*, dst. Struktur-struktur ini juga dibuat dengan HTML.

Dalam proses pembuatan website, penempatan dari struktur ini dikenal dengan pembuatan layout, yakni bagaimana tampilan website yang akan dibuat. Apakah nantinya logo website ditempatkan di sebelah kiri atau di bagian tengah, apakah website memiliki 1 atau 3 sidebar, bagaimana tampilan dari footer, dll.

Sebagai contoh kasus, saya akan merancang sebuah tampilan layout halaman seperti gambar dibawah ini:



Gambar: Contoh Layout Halaman

Walaupun saya tidak menambahkan keterangan apapun, tapi anda sudah bisa menebak fungsi dari setiap kotak dari gambar di atas. Ini adalah sebuah layout lengkap dari website modern yang sering kita jumpai.

Pada bagian atas terdapat header yang terdiri dari logo dan menu navigasi. Di bagian tengah terdapat konten utama (biasanya berupa artikel) dan sebuah sidebar disebelah kanan. Di bagian bawah adalah kolom footer yang digunakan untuk menampilkan fungsi tambahan dan teks copyright.

16.2 Membuat Layout dengan Div Element

Untuk membuat tampilan layout atau struktur halaman seperti gambar sebelumnya, kita harus mengkombinasikan HTML dengan CSS. HTML digunakan untuk membuat struktur, sedangkan CSS digunakan untuk membuat ‘style’ halaman.



Khusus untuk kode CSS yang digunakan, saya tidak akan membahasnya terlalu banyak, karena kita hanya fokus ke dalam kode HTML. Secara garis besar, kode CSS wajib digunakan adalah property `float`.

Sudah menjadi ‘standar’ di kalangan web designer, menggunakan tag `<div>` untuk merancang tampilan layout tersebut. Tag `<div>` (dan ‘sobatnya’, tag ``) adalah tag HTML yang ‘tidak bermakna’, atau sebagai tag ‘generic’ yang bisa digunakan untuk keperluan apa saja. Tag `<div>` bertipe **block level element**, sehingga cocok digunakan untuk membagi-bagi layout halaman.

Sebagai contoh, untuk membuat bagian header tempat logo dan menu navigasi berada, kita bisa menggunakan kode HTML sebagai berikut:

```
<div>
  
  <ul>
    <li><a href="home.html">Home</a></li>
    <li><a href="pendaftaran.html">Pendaftaran</a></li>
    <li><a href="blog.html">Blog</a></li>
    <li><a href="about_us.html">About Us</a></li>
  </ul>
</div>
```

Saya menggunakan tag `<div>` untuk menampung seluruh bagian header. Untuk logo, kita tinggal menggunakan tag ``, sedangkan untuk bagian menu, saya menggunakan tag `` dan `` untuk membuat list menu. Di dalam tag `` ini disisipkan link dengan tag `<a>`. Ini adalah cara penulisan menu yang umum digunakan.

Jika anda bertanya, kenapa menggunakan list? Bukankah nanti tampilan akhirnya akan terdapat tanda bulatan di depan list? Anda benar!, Tag `` dan tag `` di atas akan menghasilkan tampilan unordered list yang secara default akan ditampilkan dengan tanda bulatan di depan list. Namun dengan CSS, kita bisa menyulap list ini menjadi menu navigasi tanpa tanda bulatan list dan ditampilkan mendatar (horizontal).

Untuk keperluan CSS dan juga JavaScript, biasanya kode HTML di atas masih ditambah dengan beberapa tag `<div>` lainnya yang akan membungkus bagian logo dan menu navigasi. Sehingga kode kita sebelumnya menjadi seperti berikut ini:

```
<div>
  <div>
    
  </div>

  <div>
    <ul>
      <li><a href="home.html">Home</a></li>
      <li><a href="pendaftaran.html">Pendaftaran</a></li>
      <li><a href="blog.html">Blog</a></li>
      <li><a href="about_us.html">About Us</a></li>
    </ul>
  </div>
</div>
```

Kode di atas mirip dengan kode kita sebelumnya, dengan tambahan dua buah tag `<div>` yang membungkus logo dan menu.

Revisi terakhir yang diperlukan untuk kode tersebut adalah penambahan identitas untuk setiap tag div. Bisa dilakukan dengan menambahkan atribut "id" pada setiap tag `<div>`. Dengan demikian kita bisa mengetahui apa fungsi dari tag-tag ini. Penambahan atribut "id" juga akan berguna sebagai selector untuk kode CSS dan JavaScript.

Berikut revisi kode programnya:

```
<div id="header">
  <div id="logo">
    
  </div>

  <div id="menu_navigasi">
    <ul>
      <li><a href="home.html">Home</a></li>
      <li><a href="pendaftaran.html">Pendaftaran</a></li>
      <li><a href="blog.html">Blog</a></li>
      <li><a href="about_us.html">About Us</a></li>
    </ul>
  </div>
</div>
```

Sekarang, setiap tag `<div>` sudah memiliki identitas masing-masing.

Mirip seperti langkah-langkah sebelumnya, bagian konten, sidebar dan footer dari halaman nantinya juga akan menggunakan tag `<div>` untuk membagi-bagi struktur halaman. Sebagai contoh lainnya, berikut adalah kode struktur untuk bagian konten atau isi utama dari website kita plus bagian sidebar:

```
<div id="konten_utama">
  <div id="artikel">
    <div id="header_artikel">
      <h1>Judul Artikel</h1>
    </div>
    <p>....isi artikel disini....</p>
    <p>....isi artikel disini....</p>
    <p>....isi artikel disini....</p>
    <p>....isi artikel disini....</p>
  </div>
</div>

<div id="sidebar">
  <div id="sidebar_1">
    <h1>Judul Sidebar 1</h1>
```

```
<div id="isi_sidebar_1">
    <p>....isi sidebar 1 disini...</p>
</div>
</div>
<div id="sidebar_2">
    <h1>Judul Sidebar 2</h1>
    <div id="isi_sidebar_2">
        <p>....isi sidebar 2 disini...</p>
    </div>
</div>
</div>
```

Seperti yang terlihat, saya menggunakan banyak tag `<div>` untuk membuat pembagian struktur halaman. Untuk bagian footer, caranya relatif sama. Semua ini adalah cara standar dan juga cara terbaik untuk membuat struktur halaman, hingga... kehadiran HTML5.

16.3 Pengertian Semantic Tag / Semantic Element HTML5

Penggunaan tag `<div>` dengan atribut id seperti contoh yang kita gunakan sebelum ini memiliki sebuah kekurangan, yakni tidak ada cara standar untuk membuat sebuah segment tertentu.

Sebagai contoh, untuk membuat bagian header, seseorang bisa menggunakan tag `<div>` dengan `id="header"`, `id="head"`, `id="top"`, atau `id="bagian_atas"`. Semuanya tidak ada yang salah, tetapi sedikit sulit untuk ‘robot’ seperti mesin pencari untuk mengambil informasi dari sebuah halaman.

Dalam kasus ideal, mesin pencari seperti google men-scan seluruh halaman untuk mencari informasi mana yang dianggap penting, mana yang kurang penting, dan bagian mana yang hanya sekedar tambahan. Tag-tag yang ‘memiliki makna’ seperti tag `<h1>` bisa dijadikan sebagai patokan. Namun ketika melihat tag `<div>`, google tidak bisa memutuskan apakah ini bagian yang penting atau hanya sekedar tambahan (seperti bagian sidebar). Karena kebutuhan inilah HTML 5 memperkenalkan beberapa tag baru yang ‘memiliki makna’.

Tag yang memiliki makna ini dalam istilah HTML 5 dikenal sebagai **semantic tag**. Beberapa diantaranya adalah `<header>`, `<article>`, `<section>`, `<nav>`, dan `<aside>`. Seperti yang anda lihat dari namanya, tag-tag ini ditujukan untuk mengganti peran tag `<div>` sebagai pembuat struktur halaman.

Walaupun dirancang memiliki makna, hampir semua semantic tag ini tidak memiliki efek tampilan bawaan. Sebagai contoh, jika anda membuat kode HTML : `<header>Ini adalah header </header>`. Tulisan “Ini adalah header” tidak akan ditampilkan dengan tebal, atau dalam ukuran besar. Tag ini tidak mengubah tampilan. Pengaturan bagaimana sebuah tag `<header>` ditampilkan diserahkan kepada CSS.

i Jika dilihat dari pengertiannya, semantic tag bukanlah hal baru di dalam HTML. Tag seperti `<p>`, `<form>`, `<h1>`, `<table>` dan `` semuanya ‘memiliki makna’, dan adalah semantic tag. HTML5 hanya menambahkan beberapa semantic tag untuk bagian yang belum memiliki tag khusus (yang selama ini dibuat dengan tag `<div>`).

Berikutnya, kita akan membahas tag-tag semantic ini dan melihat cara penggunaannya.

16.4 Header Element

Tag `<header>` ditujukan untuk... bagian header :)

Tag ini berfungsi sebagai penanda bagian ‘atas’ atau head dari sebuah halaman atau artikel. Anda akan langsung menebak bahwa tag ini cocok untuk menggantikan tag `<div id="header">` dari contoh kode kita. Dan tentunya ini tidak salah. Berikut adalah modifikasi bagian header dari kode HTML sebelumnya:

```
<header>
  <div id="logo">
    
  </div>

  <div id="menu_navigasi">
    <ul>
      <li><a href="home.html">Home</a></li>
      <li><a href="pendaftaran.html">Pendaftaran</a></li>
      <li><a href="blog.html">Blog</a></li>
      <li><a href="about_us.html">About Us</a></li>
    </ul>
  </div>
</header>
```

Selain itu, tag `<header>` juga cocok digunakan untuk bagian lain. Sebagai contoh, dalam sebuah artikel, Judul artikel kadang juga memiliki sub judul, dan kemudian diikuti dengan meta-data seperti tanggal artikel, kategori artikel, tag dan nama penulis. Seluruh bagian judul ini juga sesuai untuk tempat tag `<header>`. Berikut contoh penggunaannya:

```
<header>
  <h1>Tutorial Memasak Telur Dadar</h1>
  <h2>menu khas sajian anak kost</h2>
  <time>15 Mei 2015</time>
  <span>Kategori: masakan</span>
  <span>Tag: anak kost, telur, telur dadar</span>
  <span>Penulis: udin syamsudin</span>
</header>
```

Selain itu, penggunaan tag `<header>` juga cocok untuk bagian yang membutuhkan pembagian struktur. Bukan tidak mungkin di bagian footer dari sebuah website juga dibuat menggunakan tag `<header>`, sebagai tempat untuk judul dari bagian footer.

16.5 Section Element

Tag `<section>` digunakan untuk membuat sebuah bagian (section) di dalam struktur halaman. Tag ini sebenarnya masih terkesan ‘generik’ karena penggunaannya tidak hanya untuk bagian tertentu. Sebagai contoh, tag ini bisa digunakan untuk ‘membungkus’ bagian konten yang terdiri dari beberapa artikel, atau untuk digunakan untuk memisahkan bagian sidebar.

Sebagai contoh, kita bisa mengganti penggunaan tag `<div id="sidebar_1">` dengan tag `<section>`. Dengan demikian sidebar terdiri dari beberapa section:

```
<div id="sidebar">
  <section>
    <h1>Judul Sidebar 1</h1>
    <div id="isi_sidebar_1">
      <p>....isi sidebar 1 disini...</p>
    </div>
  </section>
  <section>
    <h1>Judul Sidebar 2</h1>
    <div id="isi_sidebar_2">
      <p>....isi sidebar 2 disini...</p>
    </div>
  </section>
</div>
```

Salah satu catatan dari spesifikasi HTML5, bahwa tag `<section>` bukanlah pengganti dari tag `<div>`. Jika kita ingin membuat struktur yang hanya berfungsi untuk ‘tempat’ CSS, tetap gunakan tag `<div>`.

16.6 Article Element

Sesuai dengan namanya, tag `<article>` cocok digunakan untuk struktur artikel. Dalam contoh kita, tag ini bisa digunakan untuk mengganti bagian `<div id="artikel">`, seperti berikut:

```
<section>
  <article>
    <header>
      <h1>Judul Artikel</h1>
    </header>
    <p>....isi artikel disini...</p>
    <p>....isi artikel disini...</p>
    <p>....isi artikel disini...</p>
    <p>....isi artikel disini...</p>
  </article>
</section>
```

Dalam spesifikasi HTML5, tag `<article>` juga bisa digunakan pada komentar user, sidebar, dan bagian dari konten yang bisa berdiri sendiri.

Perbedaan antara Section dan Article Element

Pada awal perkenalannya, tag `<section>` dan tag `<article>` sering menjadi perdebatan. Kapan sebaiknya menggunakan tag `<section>` dan kapan menggunakan tag `<article>`.

Secara garis besar, tag `<article>` adalah tag `<section>` untuk tujuan yang lebih spesifik. Jika bagian dari konten bisa berdiri sendiri dan tidak tergantung dengan bagian lain, maka gunakan tag `<article>`. Namun jika konten tersebut adalah bagian dari struktur yang lebih besar, gunakan tag `<section>`.

Kedua tag ini juga bisa digunakan secara bersama-sama. Sebagai contoh, untuk artikel yang panjang, kita bisa memecahnya menjadi beberapa section. Berikut contoh kode HTML-nya:

```
<article>
  <h1>Jenis-jenis Telur </h1>
  <p>Telur terbagi menjadi beberapa jenis:</p>

  <section>
    <h2>Telur Ayam Kampung</h2>
    <p>Penjelasan tentang telur ayam kampung disini</p>
  </section>

  <section>
    <h2>Telur Itik</h2>
    <p>Penjelasan tentang telur itik disini</p>
  </section>
</article>
```

Demikian pula jika dalam sebuah halaman terdapat beberapa artikel, kita bisa mengumpulkan-nya ke dalam sebuah section:

```

<section>
  <h1>Articles tentang: Telur</h1>

  <article>
    <h2>Jenis-jenis Telur</h2>
    <p>Telur terbagi ke dalam beberapa jenis, yakni.....</p>
  </article>

  <article>
    <h2>Manfaat Makan Telur</h2>
    <p>Telur sangat bermanfaat bagi tubuh, seperti....</p>
  </article>

  <article>
    <h2>Cara Memasak Telur</h2>
    <p>Telur bisa dimasak menjadi masakan yang sangat enak, contohnya.....</p>
  </article>
</section>

```

16.7 Nav Element

Tag `<nav>` digunakan untuk mengelompokkan beberapa link yang membuat sebuah menu navigasi. Dengan kata lain, `<nav>` cocok sebagai ‘container’ untuk menu.

Dalam kode kita sebelumnya, untuk menu navigasi saya menggunakan tag `<div id="menu_navigasi">`. Kali ini kita akan mengubahnya menjadi tag `<nav>`:

```

<nav>
  <ul>
    <li><a href="home.html">Home</a></li>
    <li><a href="pendaftaran.html">Pendaftaran</a></li>
    <li><a href="blog.html">Blog</a></li>
    <li><a href="about_us.html">About Us</a></li>
  </ul>
</nav>

```

Selain untuk menu utama, tag `<nav>` juga bisa digunakan untuk menu navigasi di footer, table of contents (daftar isi), *pagination*, dan *breadcrumbs*.

16.8 Aside Element

Tag `<aside>` berfungsi untuk menyatakan bagian dari konten yang bukan merupakan bagian dari konten utama, namun masih berhubungan. Dalam implementasinya, tag `<aside>` paling jelas digunakan untuk bagian sidebar.

Berikut adalah modifikasi dari kode kita sebelumnya yang menggunakan `<div id="sidebar">` untuk menandai sidebar:

```
<aside>
  <section>
    <h1>Judul Sidebar 1</h1>
    <div id="isi_sidebar_1">
      <p>....isi sidebar 1 disini...</p>
    </div>
  </section>
  <section>
    <h1>Judul Sidebar 2</h1>
    <div id="isi_sidebar_2">
      <p>....isi sidebar 2 disini...</p>
    </div>
  </section>
</aside>
```

Selain untuk sidebar, tag `<aside>` juga bisa diletakkan di dalam konten utama, yang ditujukan untuk memberi keterangan tambahan.

16.9 Main Element

Tag `<main>` digunakan untuk menandai bagian mana yang berperan sebagai konten utama. Dalam setiap halaman sebaiknya hanya terdapat 1 tag `<main>`. Web browser khusus seperti screen reader yang digunakan untuk pengguna disabilitas bisa mengabaikan konten lain dan langsung masuk ke bagian `<main>`.

Dalam contoh kita, tag ini bisa digunakan sebagai pengganti tag `<div id="konten_utama">`

```
<main>
  <article>
    <header>
      <h1>Judul Artikel</h1>
    </header>
    <p>....isi artikel disini...</p>
    <p>....isi artikel disini...</p>
    <p>....isi artikel disini...</p>
    <p>....isi artikel disini...</p>
  </article>
</main>
```

16.10 Footer Element

Fungsi tag ini cukup jelas, dimana tag `<footer>` digunakan untuk menampung bagian footer dari sebuah halaman website. Berikut contohnya:

```
<footer>
  <p>© 2015 Dunia Ilkom. All Rights Reserved.</p>
</footer>
```

Sama seperti tag `<header>`, tag `<footer>` tidak hanya digunakan untuk bagian akhir halaman, namun juga untuk bagian akhir dari artikel. Biasanya bagian ini berisi biodata tentang penulis artikel dan link-link yang berkaitan dengan artikel yang dibahas. Seperti contoh berikut ini:

```
<article>
  <header>
    <h1>Judul Artikel</h1>
  </header>
  <p>....isi artikel disini...</p>
  <p>....isi artikel disini...</p>
  <p>....isi artikel disini...</p>
  <p>....isi artikel disini...</p>
  <footer>
    <p>Artikel ini ditulis oleh...</p>
  </footer>
</article>
```

16.11 Hgroup Element

Tag `<hgroup>` pada awalnya dirancang sebagai penampung untuk beberapa judul artikel seperti `<h1>`, `<h2>`, dst. Namun pada spesifikasi final HTML5, tag ini **resmi di hapus**, dan fungsinya ditambahkan kepada tag `<header>`. Saya sengaja membahasnya karena dalam beberapa sumber masih ada yang memasukkan tag ini ke dalam semantic tag HTML5.

16.12 Contoh Layout Halaman HTML5

Sebagai penutup untuk bab tentang HTML5 Struktur, saya akan memberikan sebuah halaman lengkap yang menggunakan hampir seluruh semantic element yang kita pelajari. Dalam contoh ini saya juga banyak menggunakan kode CSS untuk mengatur tampilan halaman agar sama persis seperti gambar yang kita lihat diawal bab.

Berikut kode HTML lengkapnya:

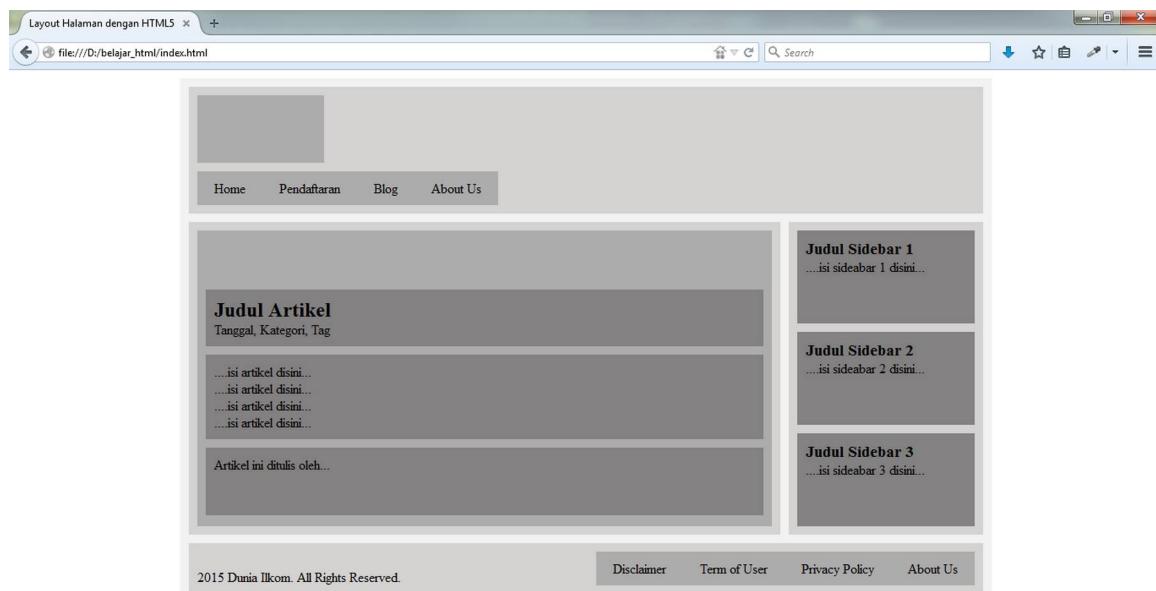
index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Layout Halaman dengan HTML5</title>
6   <style>
7     h1, p {
8       margin: 0px;
9     }
10    #container {
11      width: 960px;
12      margin: 10px auto;
13      background-color: #f2f2f2;
14      height: 620px;
15    }
16    #container>header{
17      width: 940px;
18      float: left;
19      margin: 10px;
20      background-color: #d5d2d2;
21      height: 150px;
22    }
23    #logo {
24      width: 150px;
25      height: 80px;
26      float: left;
27      background-color: #adacac;
28      margin: 10px;
29      clear: both;
30    }
31    nav {
32      float: left;
33      background-color: #adacac;
34      margin: 0 10px;
35      clear: both;
36    }
37    nav ul {
38      list-style: none;
39      margin: 0;
40      padding: 0;
41    }
42    nav ul li {
43      float: left;
44    }
```

```
45  nav ul a {
46    text-decoration: none;
47    color: black;
48    padding: 10px 20px;
49    height: 20px;
50    display: block;
51  }
52  nav ul a:hover {
53    color: white;
54    background-color: #838181;
55  }
56  main{
57    width: 700px;
58    float: left;
59    margin: 0px 10px 10px 10px;
60    background-color: #d5d2d2;
61    height: 370px;
62  }
63  article {
64    width: 680px;
65    float: left;
66    margin: 10px;
67    background-color: #adacac;
68    height: 350px;
69  }
70  article header, article section, article footer {
71    background-color: #838181;
72    width: 640px;
73    padding: 10px;
74    margin: 10px 10px 0 10px;
75    float: left;
76    clear: both;
77  }
78  article header{
79    margin-top: 70px;
80  }
81  aside{
82    width: 230px;
83    float: left;
84    background-color: #d5d2d2;
85    height: 370px;
86  }
87  aside section {
88    background-color: #838181;
89    width:190px;
90    padding: 10px;
```

```
91     margin: 10px 10px 0 10px;
92     height: 90px;
93     float: left;
94     clear: both;
95 }
96 footer {
97     width: 940px;
98     float: left;
99     margin: 0px 10px 10px 10px;
100    background-color: #d5d2d2;
101    height: 60px;
102}
103 footer div {
104    margin: 30px 10px;
105}
106 footer p {
107    float: left;
108}
109 footer nav {
110    float: right;
111    margin: -40px 0 0 0;
112}
113 </style>
114 </head>
115 <body>
116 <div id="container">
117 <header>
118     <div id="logo">
119     </div>
120     <nav>
121         <ul>
122             <li><a href="home.html">Home</a></li>
123             <li><a href="pendaftaran.html">Pendaftaran</a></li>
124             <li><a href="blog.html">Blog</a></li>
125             <li><a href="about_us.html">About Us</a></li>
126         </ul>
127     </nav>
128 </header>
129
130 <main>
131     <article>
132         <header>
133             <h1>Judul Artikel</h1>
134             <p>Tanggal, Kategori, Tag</p>
135         </header>
136         <section>
```

```
137      <p>....isi artikel disini...</p>
138      <p>....isi artikel disini...</p>
139      <p>....isi artikel disini...</p>
140      <p>....isi artikel disini...</p>
141  </section>
142  <footer>
143      <p>Artikel ini ditulis oleh...</p>
144  </footer>
145  </article>
146 </main>
147
148 <aside>
149  <section>
150      <h1>Judul Sidebar 1</h1>
151      <p>....isi sidebar 1 disini...</p>
152  </section>
153  <section>
154      <h1>Judul Sidebar 2</h1>
155      <p>....isi sidebar 2 disini...</p>
156  </section>
157  <section>
158      <h1>Judul Sidebar 3</h1>
159      <p>....isi sidebar 3 disini...</p>
160  </section>
161 </aside>
162
163 <footer>
164  <div>
165      <p> 2015 Dunia Ilkom. All Rights Reserved.</p>
166  <nav>
167      <ul>
168          <li><a href="disclaimer.html">Disclaimer</a></li>
169          <li><a href="term_of_use.html">Term of User</a></li>
170          <li><a href="privacy_policy.html">Privacy Policy</a></li>
171          <li><a href="about_us.html">About Us</a></li>
172      </ul>
173  </nav>
174  </div>
175 </footer>
176 </div>
177 </body>
178 </html>
```



Gambar: Hasil Layout Halaman dengan HTML5

Kode CSS untuk halaman di atas cukup banyak, namun anda hanya perlu fokus dengan tag-tag HTML yang digunakan dan bagaimana membuat struktur halaman menggunakan HTML5.

17. HTML 5 Update

Bab ini saya khususkan untuk membahas fitur terbaru yang ditambahkan ke dalam HTML. Sejak dirilisnya **HTML 5** pada 2014, W3C sebagai organisasi yang bertanggung jawab dalam pengembangan HTML sudah mengeluarkan 2 buah update, yakni **HTML 5.1** pada November 2016 dan **HTML 5.2** pada Desember 2017.

Masing-masing versi HTML ini membawa berbagai fitur baru seperti perbaikan bug, penambahan **HTML 5 API**, serta penambahan atribut, element dan tag baru.

Mayoritas update memang lebih banyak ke fitur advanced seperti **HTML 5 API** dan saya tidak akan membahas tentang hal ini karena harus melibatkan JavaScript. Kita akan fokus ke penambahan atribut, element dan tag baru saja.

17.1 Update HTML 5.1

Standar **HTML 5.1**¹ di rilis sebagai rekomendasi pada bulan November 2016 dan diupdate pada 3 October 2017. Berikut ringkasan dari penambahan atribut, element dan tag baru pada versi HTML ini:

Fitur ditambah:

- <picture> element dan atribut srcset untuk membuat gambar responsive.
- <details> dan <summary> element untuk membuat informasi tambahan yang bersifat opsional (bisa dibaca atau dilewati).
- <menuitem> element dan atribut type="context" untuk membuat *context menu* ke dalam web browser (menu yang keluar saat klik kanan).

Fitur dihapus:

- Atribut usemap untuk <object> element.
- Atribut form untuk <label> element.
- Atribut multiple untuk <input type="range"> element.
- Atribut hreflang dan type untuk <area> element.
- Larangan peletakan posisi <tfoot> sebelum <tbody>.

Daftar lengkap terkait perubahan di **HTML 5.1** bisa diakses ke **HTML 5.1 Changes**². Kita akan bahas beberapa diantaranya.

¹<https://www.w3.org/TR/html51/>

²<https://www.w3.org/TR/html51/changes.html#changes-since-51>

17.2 Picture Element

Picture element, atau tag `<picture>` merupakan sebuah *container* atau penampung dari beberapa gambar. Gambar itu sendiri nantinya ditempatkan ke dalam tag `<source>` beserta atribut `srcset` serta sebuah tag ``.

Tag `<picture>` bertujuan untuk membuat **gambar responsive**, dimana kita bisa menampilkan gambar yang berbeda-beda tergantung ukuran layar pengguna. Efek yang di dapat kurang lebih sama seperti **media query** di CSS (yang akan dibahas di buku CSS Uncover).

Agar pengertian ini bisa lebih jelas, langsung saja kita bahas dengan contoh praktek. Sebelumnya, saya sudah menyiapkan 3 buah gambar:

- `coffee_cup_small.jpg`, dengan lebar gambar 350px serta tinggi gambar 300px.
- `coffee_cup_medium.jpg`, dengan lebar gambar 500px serta tinggi gambar 400px.
- `coffee_cup_big.jpg`, dengan lebar gambar 850px serta tinggi gambar 450px.

Berikut tampilan ketiga gambar ini:



Gambar: Tiga gambar `coffee_cup`

Pada dasarnya, gambar `coffee_cup_small.jpg` dan `coffee_cup_medium.jpg` adalah hasil pemotongan (crop) dari gambar `coffee_cup_big.jpg`. Ketiganya juga sudah saya sertakan di dalam file `belajar_html.zip`.

Sebagai contoh pertama, saya akan “membungkus” tag `` dengan tag `<picture>`:

`01.picture_img.html`

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Picture Element</title>
6 </head>
7 <body>
8   <picture>

```

```
9     
10    </picture>
11  </body>
12 </html>
```

Hasilnya, gambar coffee_cup_big.jpg akan tampil di web browser. Tanpa atau dengan tag <picture> tidak akan terlihat perbedaan secara visual. Disini tag <picture> hanya berperan sebagai container (penampung) dari tag .

Selanjutnya, saya akan tambah sebuah tag <source> dengan atribut srcset ke dalam tag <picture> ini:

02.picture_max-width.html

```
1 <picture>
2   <source srcset="coffee_cup_medium.jpg" media="(max-width: 850px)">
3   
4 </picture>
```

Perhatikan isi dari tag <picture>, di dalamnya terdapat 2 buah tag: <source> dan .

Tag <source> disini berfungsi untuk memberikan gambar alternatif. Dengan syarat, web browser tersebut sudah mendukung tag ini. Jika tidak, yang ditampilkan adalah gambar yang berasal dari tag .

Tag <source> sendiri memiliki 2 buah atribut: srcset dan media. Atribut srcset berfungsi sama seperti atribut src dari tag , yakni sebagai tempat untuk menulis alamat gambar yang akan ditampilkan. Dalam hal ini saya ingin menampilkan gambar coffee_cup_medium.jpg.

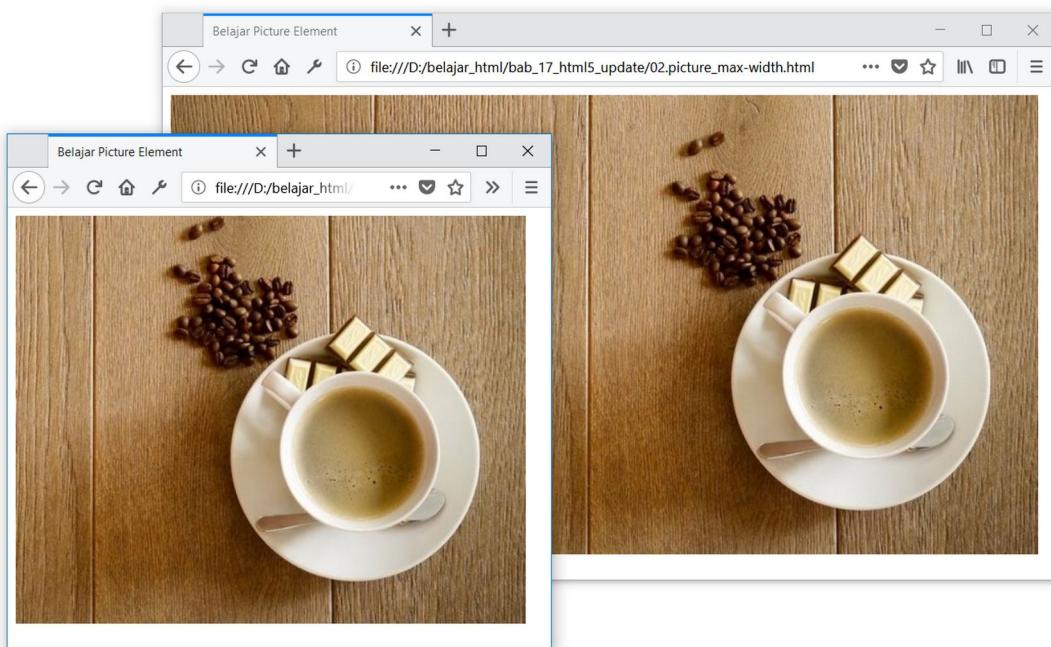
Atribut media berfungsi layaknya **media query** di CSS. Yakni sebuah kondisi atau syarat berapa lebar layar web browser supaya gambar yang ada di dalam atribut srcset bisa tampil.

Dalam contoh ini saya menulis media="(max-width: 850px)". Artinya, gambar coffee_cup_medium.jpg akan ditampilkan **jika ukuran lebar jendela web browser maksimal sebesar 850 pixel**. Apabila lebar web browser lebih besar dari 850 pixel, gambar tidak akan terlihat. Dengan kata lain, gambar baru tampil ketika lebar jendela web browser lebih kecil dari 850 pixel.

Bagaimana jika lebar web browser ternyata lebih dari 850 pixel? Karena ini tidak memenuhi syarat max-width: 850px, maka yang tampil adalah gambar coffee_cup_big.jpg milik tag .

Dalam prakteknya, lebar jendela web browser ini tidak lain adalah ukuran layar pengguna. Jika kita membuka halaman web dari laptop atau komputer, biasanya lebar layar lebih dari 1000 pixel. Namun untuk smartphone atau tablet, lebar layar bervariasi mulai dari 300 pixel hingga 700 pixel.

Untuk melihat efek dari tag <picture> ini, silahkan buka web browser lalu perkecil ukuran jendela web browser.



Gambar: Gambar akan berganti ketika ukuran web browser kecil dari 850px

Saat kode diatas dijalankan pertama kali, yang tampil adalah gambar `coffee_cup_big.jpg`. Ketika lebar jendela web browser diperkecil menjadi 850 pixel atau dibawahnya, gambar akan berganti dengan `coffee_cup_medium.jpg`

Kita juga bisa menambah beberapa tag `<source>` untuk membuat kondisi lain. Sebagai contoh, saya ingin membuat kondisi lain yakni ketika jendela web browser lebih kecil dari 500 pixel, tampilkan gambar `coffee_cup_small.jpg`. Berikut kode programnya:

03.picture_max-width_2.html

```

1 <picture>
2   <source srcset="coffee_cup_small.jpg" media="(max-width: 500px)">
3   <source srcset="coffee_cup_medium.jpg" media="(max-width: 850px)">
4   
5 </picture>

```

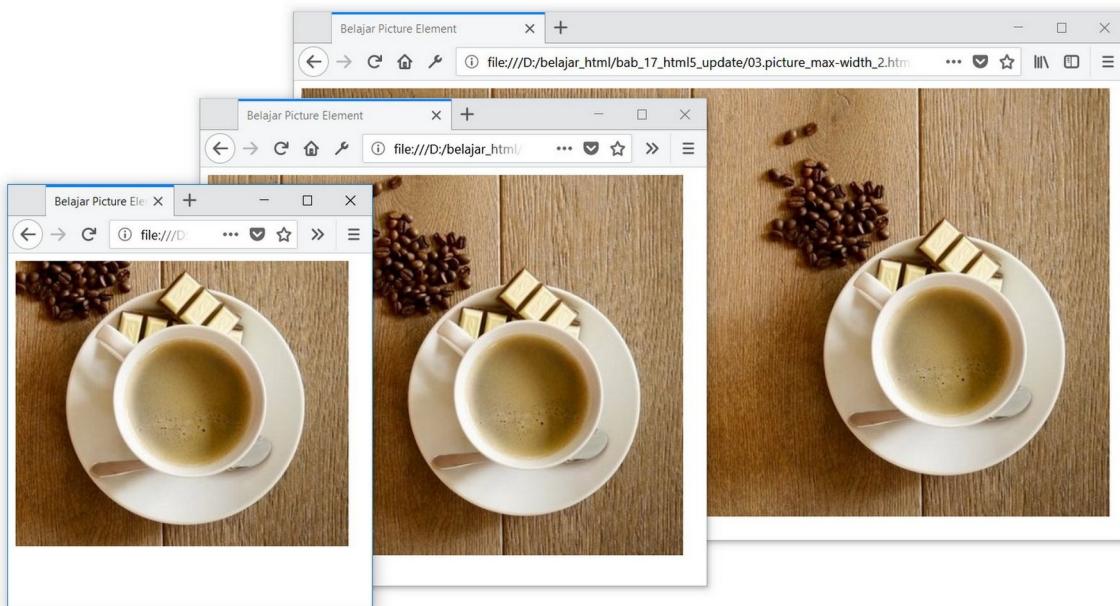
Sekarang terdapat 2 buah tag `<source>`. Perhatikan isi dari atribut media untuk masing-masing tag ini.

Tag `<source>` pertama bisa dibaca: Jika lebar jendela web browser lebih kecil dari 500 pixel, maka tampilkan gambar `coffee_cup_small.jpg`. Kondisi ini didapat dari tambahan atribut `media="(max-width: 500px)"`.

Tag `<source>` kedua bisa dibaca: Jika lebar jendela web browser lebih kecil dari 850 pixel, maka tampilkan gambar `coffee_cup_medium.jpg`. Kondisi ini didapat dari tambahan atribut `media="(max-width: 850px)"`.

Terakhir, tag `` akan ditampilkan jika kedua kondisi diatas tidak terpenuhi.

Untuk melihat efeknya, kembali buka web browser lalu ubah ukuran jendela:



Gambar: Akan terlihat 3 kali pergantian gambar ketika ukuran jendela web browser diperkecil / di perbesar

Ketika ukuran jendela web browser diperkecil atau diperbesar, gambar akan berganti-ganti. Inilah fungsi dari tag <picture> di HTML 5.1

Yang juga harus diperhatikan, proses pembacaan kondisi dilakukan dari atas ke bawah (baris pertama lebih dulu, lalu lanjut ke baris kedua, dst). Ketika web browser menemukan kondisi yang sesuai, maka tag <source> dibawahnya tidak akan diproses lagi. Sehingga selalu dahlulukan penulisan tag <source> untuk kondisi yang paling kecil.

Sebagai alternatif dari kondisi `max-width`, kita juga bisa menggunakan kondisi `min-width`. Disini kondisi logikanya dibalik. Jika ditulis `media="(min-width: 850px)"`, maka kondisi tersebut akan terpenuhi jika lebar jendela web browser minimal 850 pixel, yang artinya lebar jendela web browser harus lebih besar dari 850px.

Berikut contoh penggunaannya:

04.picture_min-width.html

```

1 <picture>
2   <source srcset="coffee_cup_big.jpg" media="(min-width: 850px)">
3   <source srcset="coffee_cup_medium.jpg" media="(min-width: 500px)">
4   
5 </picture>

```

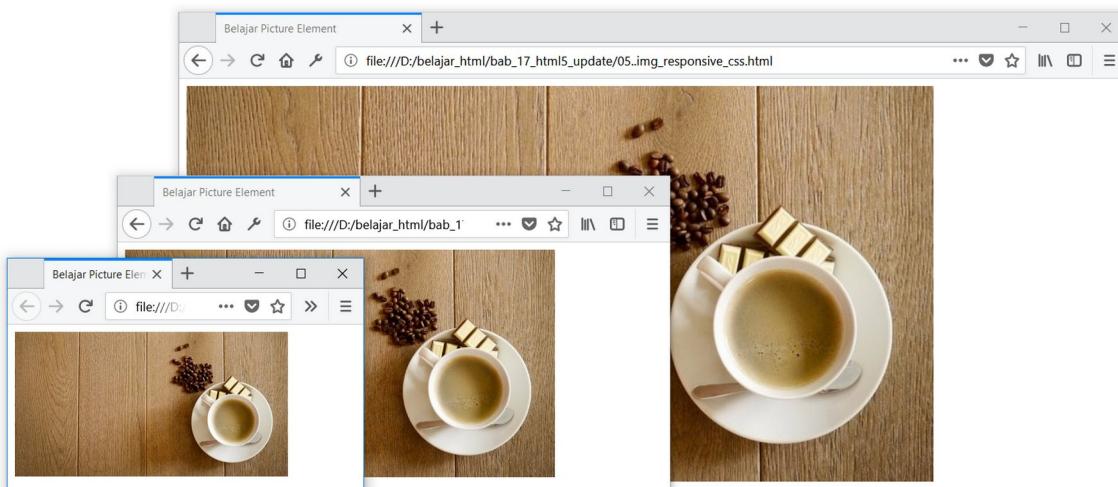
Hasil yang didapat akan sama seperti contoh kode program yang menggunakan `max-width` sebelum ini.

Selain menggunakan tag <picture>, sebenarnya kita juga bisa membuat gambar responsive tanpa harus mengganti gambar. Triknya, gunakan **ukuran relatif** seperti persen untuk lebar gambar:

05.img_responsive_css.html

```
1 
```

Disini saya menggunakan sedikit kode CSS, yakni atribut `style="width:80%"`. Atribut ini akan membuat ukuran gambar berpatokan kepada lebar web browser. Berapapun lebar web browser, gambar akan ditampilkan sebesar 80%. Berikut hasilnya:



Gambar: Gambar responsive dengan width: 80%

Dengan memperbesar dan memperkecil ukuran web browser, lebar dan tinggi gambar juga akan menyesuaikan.

Terakhir, kita bisa menggabung penggunaan atribut `srcset` secara langsung ke dalam tag ``, yakni tidak lagi menggunakan tag `<source>`:

06.picture_charset

```
1 <img srcset = "coffee_cup_small.jpg 500w,
2                  coffee_cup_medium.jpg 850w,
3                  coffee_cup_big.jpg 1400w"
4 src = "coffee_cup_small.jpg" alt="Secangkir Kopi">
```

Isi dari atribut `srcset` terdiri dari 3 buah gambar yang dipisah dengan tanda koma. Diakhir penulisan setiap gambar terdapat kode tambahan seperti `500w`, `850w`, dan `1400w`. Ini adalah *image's inherent width*, dimana kita menginformasikan kepada web browser ukuran dari setiap gambar.

Hasil dari kode diatas akan mirip seperti penggunaan tag `<picture>` dengan `<source>`. Hanya saja ukuran gambar juga akan berubah-ubah mirip seperti hasil `style="width:60%"`. Silahkan anda coba langsung agar bisa melihat hasilnya.

17.3 Details dan Summary Element

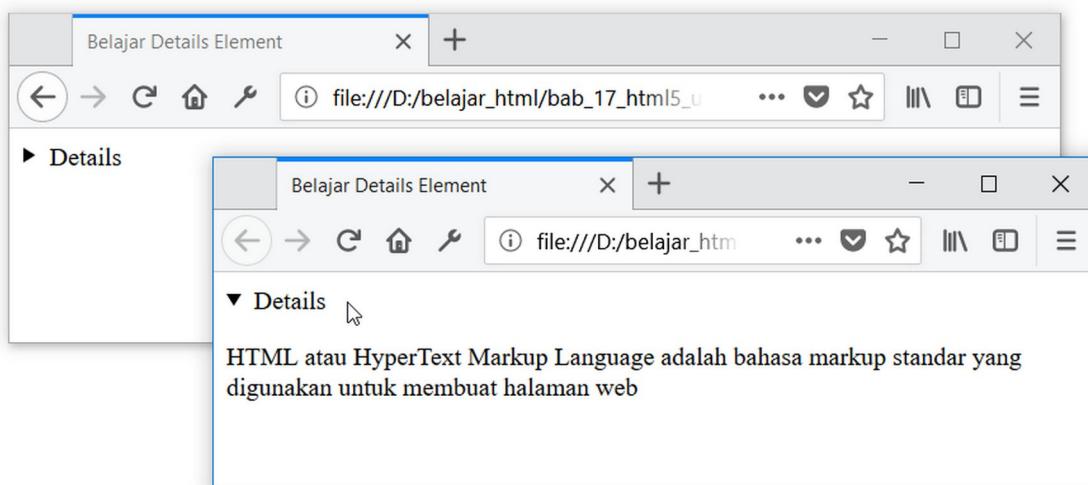
Tag <details> dan tag <summary> ditambahkan ke dalam HTML 5.1 untuk membuat efek teks tersembunyi dan baru muncul ketika kita men-klik judul teks tersebut.

Fitur seperti ini sebenarnya sudah sering dipakai, hanya saja kita harus menggunakan JavaScript untuk membuatnya. Di dalam **jQuery UI** (sebuah library JavaScript), efek ini dikenal juga dengan sebutan [Accordion](#)³.

Untuk membuat fitur ini, kita hanya butuh sebuah tag <details> yang di dalamnya berisi teks seperti contoh berikut:

07.details_element.html

```
1 <details>
2   <p>HTML atau HyperText Markup Language adalah bahasa markup standar yang
3     digunakan untuk membuat halaman web</p>
4 </details>
```



Gambar: Penggunaan tag <details> di HTML 5.1

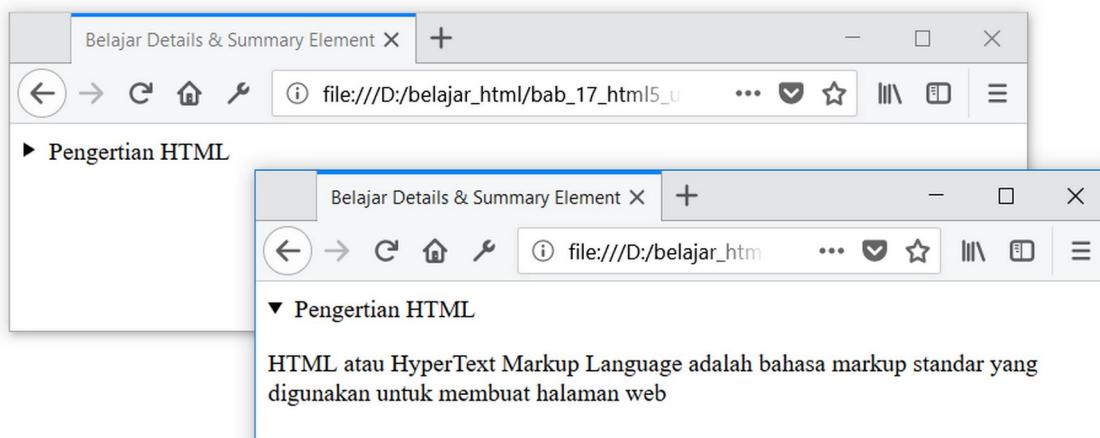
Seluruh teks yang ditempatkan ke dalam tag <details> akan tersembunyi secara default. Di dalam web browser hanya terlihat tanda panah hitam ke arah kanan dan judul teks Details. Ketika di klik, isi teks akan ditampilkan.

Bagaimana cara mengganti judul Details ini dengan teks lain? Inilah fungsi dari tag <summary> yang contoh penggunaannya sebagai berikut:

³<https://jqueryui.com/accordion>

08.details_summary_element.html

```
1 <details>
2   <summary>Pengertian HTML</summary>
3   <p>HTML atau HyperText Markup Language adalah bahasa markup standar yang
4     digunakan untuk membuat halaman web</p>
5 </details>
```



Gambar: Penggunaan tag `<details>` dan `<summary>` di HTML 5.1

Agar efek ini bisa berjalan, tag `<summary>` harus menjadi element pertama di dalam tag `<details>`. Istilah teknisnya, tag `<summary>` harus menjadi first child dari `<details>`.

Tag `<details>` memiliki sebuah atribut `open` yang jika ditambahkan, teks akan langsung tampil (tidak perlu di klik terlebih dahulu):

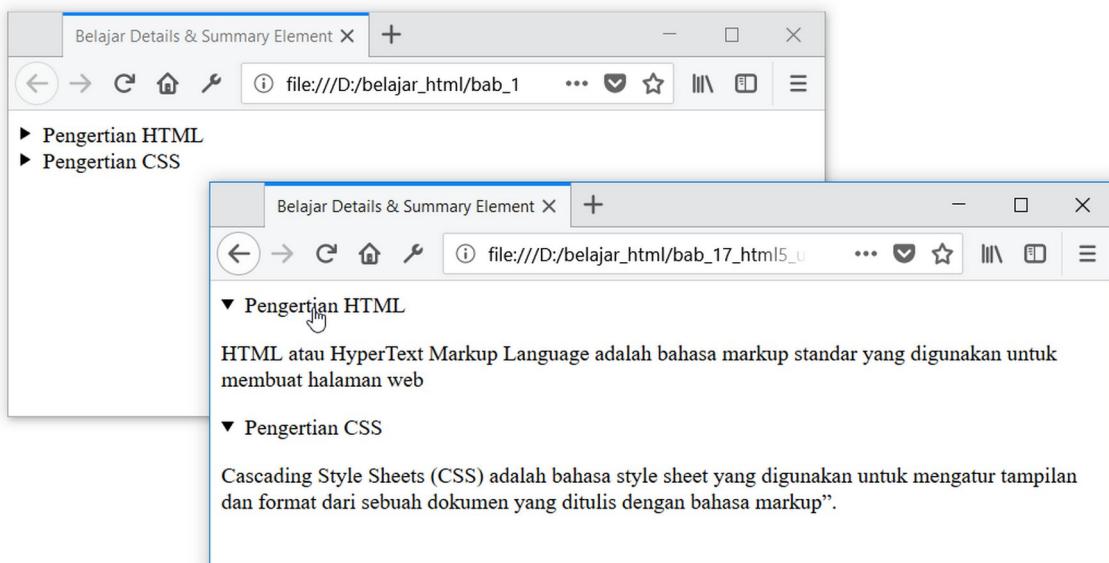
09.details_element_open.html

```
1 <details open>
2   <summary>Pengertian HTML</summary>
3   <p>HTML atau HyperText Markup Language adalah bahasa markup standar yang
4     digunakan untuk membuat halaman web</p>
5 </details>
```

Bagaimana jika kita ingin membuat lebih dari 1 teks tersembunyi? cukup dengan membuat beberapa buah tag `<details>`:

10.details_summary_element_2.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Details & Summary Element</title>
6   <style>
7     summary:hover {
8       cursor: pointer;
9     }
10  </style>
11 </head>
12 <body>
13 <details>
14   <summary>Pengertian HTML</summary>
15   <p>HTML atau HyperText Markup Language adalah bahasa markup standar yang
16     digunakan untuk membuat halaman web</p>
17 </details>
18 <details>
19   <summary>Pengertian CSS </summary>
20   <p>Cascading Style Sheets (CSS) adalah bahasa style sheet yang digunakan
21     untuk mengatur tampilan dan format dari sebuah dokumen yang ditulis
22     dengan bahasa markup".</p>
23 </details>
24 </body>
25 </html>
```



Gambar: Tampilan 2 buah tag <details>

Selain membuat 2 buah tag <details>, saya juga menambah sedikit kode CSS agar ketika cursor mouse berada dia atas judul teks, tampil gambar tangan. Ini untuk menegaskan bahwa judul tersebut bisa diklik. Efek ini didapat dari kode CSS `summary:hover { cursor: pointer; }`.

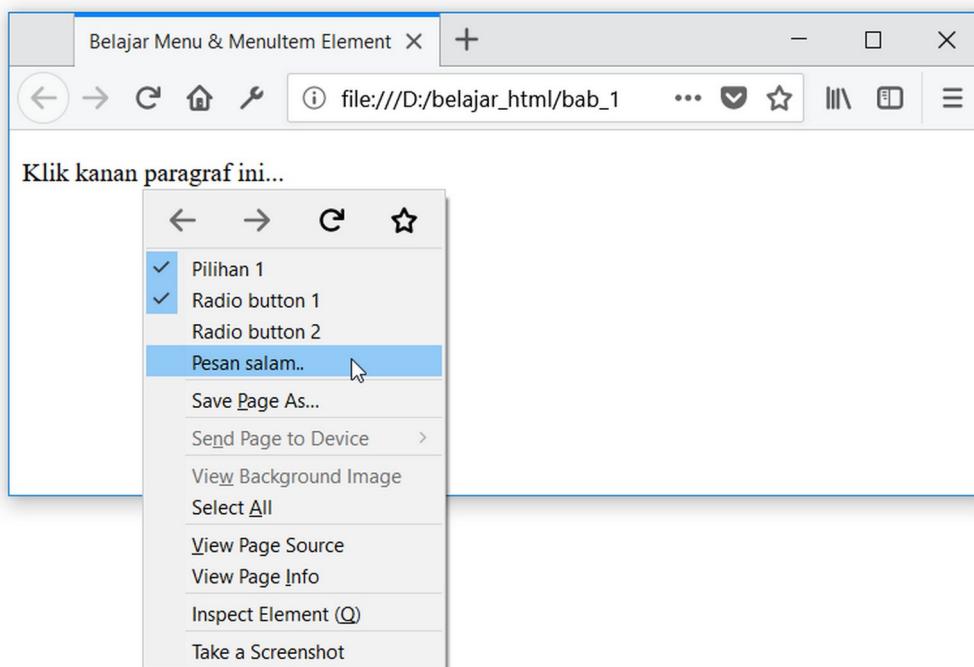
17.4 Menu dan MenuItem Element

Tag <menu> dan tag <MenuItem> diperkenalkan pada HTML 5.1. Fungsinya untuk membuat *context menu*, yakni menu yang tampil ketika kita men-klik kanan sebuah element. Fitur ini sebetulnya cukup menjanjikan, akan tetapi W3C memutuskan untuk menghapusnya di HTML 5.2.

Kedua element ini juga hanya bisa dicoba di web browser **Mozilla Firefox** (tidak bisa berjalan di web browser lain, termasuk Google Chrome dan Opera). Berikut contoh penggunaannya:

11.menuitem_element.html

```
1 <body>
2   <p contextmenu="popup-menu">
3     Klik kanan paragraf ini...
4   </p>
5
6   <menu type="context" id="popup-menu">
7     <MenuItem type="checkbox" checked="true">
8       Pilihan 1
9     </MenuItem>
10
11    <MenuItem type="radio" name="group1" checked="true">
12      Radio button 1
13    </MenuItem>
14
15    <MenuItem type="radio" name="group1">
16      Radio button 2
17    </MenuItem>
18
19    <MenuItem type="command" label="Pesan salam.." onclick="alert('Hello..')">
20      Alert
21    </MenuItem>
22  </menu>
23
24 </body>
```



Gambar: Terdapat tambahan menu ketika paragraf diatas di klik kanan

Pertama, kita harus membuat sebuah element yang akan menjadi object, yakni element yang nantinya akan diklik kanan. Dalam contoh diatas, saya membuat paragraf `<p contextmenu="popup-menu">`. Atribut `contextmenu` disini nantinya akan bersesuaian dengan atribut `id` dari tag `<menu>`.

Setelah paragraf, saya membuat sebuah tag `<menu type="context" id="popup-menu">`. Atribut `type="context"` dipakai untuk memberitahu web browser bahwa menu yang ada akan dipakai sebagai *context menu*. Atribut `id="popup-menu"` merupakan identitas pengenal dan harus bersesuaian dengan element yang menjadi object dari klik kanan. Dalam contoh ini, akan bersesuaian dengan tag `<p contextmenu="popup-menu">` sebelumnya.

Di dalam tag `<menu>`, terdapat beberapa tag `<menuitem>`. Tag `<menuitem>` inilah yang nantinya muncul ketika element di klik kanan.

Terdapat 3 jenis menu yang bisa dipakai. Pengaturan ini ditulis menggunakan atribut `type` dari tag `<menuitem>`:

- `<menuitem type="checkbox">`
- `<menuitem type="radio">`
- `<menuitem type="command">`

Tag `<menuitem type="checkbox">` dipakai untuk membuat menu yang mirip checkbox pada form. Dimana pilihannya berupa ceklist yang bisa dipilih atau tidak.

Tag `<menuitem type="radio">` dipakai untuk membuat menu yang mirip radio button pada form, dimana biasanya berpasangan dengan menu lain tapi hanya bisa dipilih salah satu saja (tidak bisa keduanya).

Tag `<menuitem type="command">` dipakai untuk membuat menu command yang biasanya berisi kode JavaScript.

Mengenai apa yang akan terjadi ketika menu-menu ini di klik, itu bukan lagi di HTML, tapi harus diproses menggunakan JavaScript.

17.5 Update HTML 5.2

Standar [HTML 5.2⁴](#) di rilis sebagai rekomendasi pada bulan Desember 2017. Berikut ringkasan dari penambahan atribut, element dan tag baru pada versi HTML ini:

Fitur ditambah:

- <dialog> element.

Fitur dihapus:

- <keygen>, <menu> and <menuitem> element.
- Atribut inputmode dan dropzone untuk <input> element.

Tidak banyak penambahan tag baru untuk **HTML 5.2**. Update terbanyak ditujukan untuk materi advanced seperti HTML API. Bahkan terdapat 3 tag yang akhirnya dihapus, yakni <keygen>, <menu> and <menuitem> element. Artinya, *context menu* yang kita pelajari sebelum ini sudah dihapus di HTML 5.2 dan sebaiknya tidak dipakai lagi.

Daftar lengkap terkait perubahan di HTML 5.2 bisa diakses ke [HTML 5.2 Changes⁵](#). Kita akan bahas tentang <dialog> element.

17.6 Dialog Element

Tag <dialog> ditujukan untuk membuat pesan dialog, yakni kotak box yang biasanya muncul ketika kita menekan tombol tertentu. Kurang lebih mirip jendela *alert* di JavaScript tapi ini dibuat dari HTML.

Berikut contoh penggunaannya:

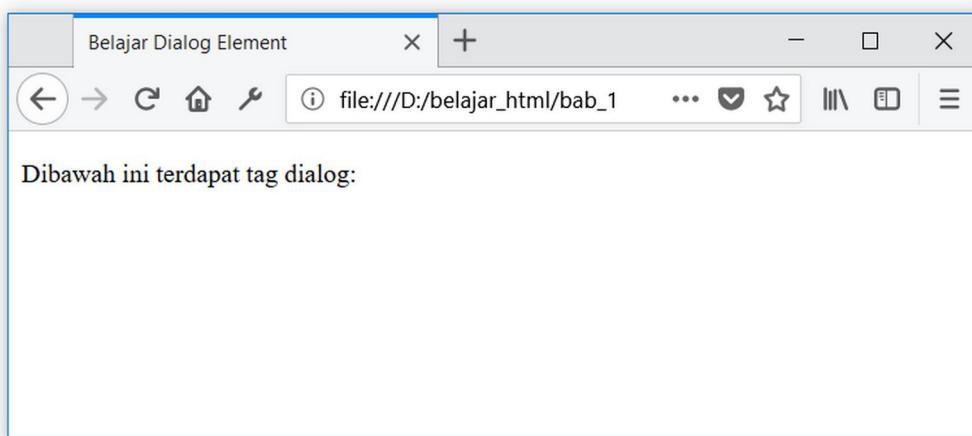
12.dialog_element.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Dialog Element</title>
6 </head>
7 <body>
8   <p>Dibawah ini terdapat tag dialog:</p>
```

⁴<https://www.w3.org/TR/html51/>

⁵<https://www.w3.org/TR/html52/changes.html#new-features>

```
9 <dialog>
10   <p>Selamat belajar HTML</p>
11 </dialog>
12 </body>
13 </html>
```



Gambar: Tampilan dari tag `<dialog>` ?

Loh, kenapa tidak tampil apa-apa? Dari kode diatas, sangat jelas terlihat bahwa saya sudah menulis tag `<dialog>` yang di dalamnya terdapat teks `<p>Selamat belajar HTML</p>`. Akan tetapi secara default isi pesan dialog ini tidak tampil.

Tampilan tag `<dialog>` baru terlihat jika tag ini ditambahkan atribut `open`:

13.dialog_element_open.html

```
1 <body>
2   <p>Dibawah ini terdapat tag dialog:</p>
3   <dialog open>
4     <p>Selamat belajar HTML</p>
5   </dialog>
6 </body>
```



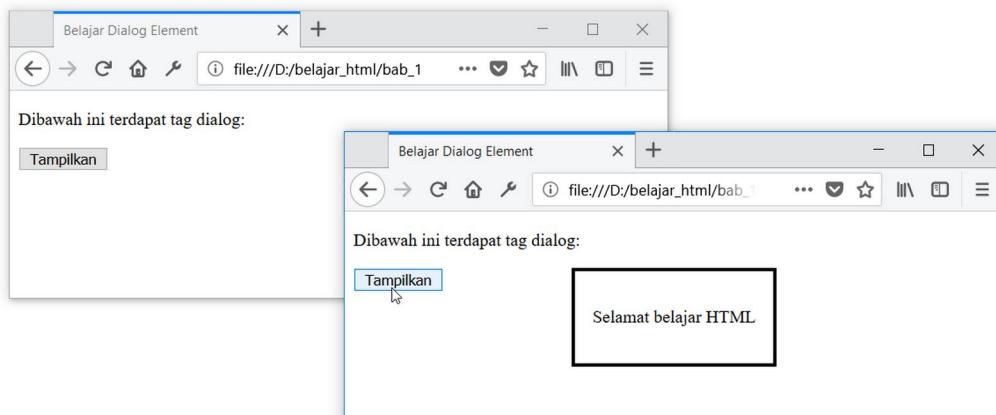
Gambar: Tampilan dari tag <dialog> dengan tambahan atribut open

Inilah tampilan dari tag <dialog>. Menggunakan kode CSS, nantinya kita bisa memodifikasi tag ini.

Pesan dialog pada dasarnya adalah sebuah efek interaksi. Misalnya pesan tersebut baru muncul ketika sebuah tombol di tekan. Ini hanya bisa dibuat menggunakan JavaScript, seperti contoh berikut:

14.dialog_element_javascript.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Belajar Dialog Element</title>
6 </head>
7 <body>
8   <p>Dibawah ini terdapat tag dialog:</p>
9   <button id="tombol">Tampilkan</button>
10
11  <dialog id="dialog">
12    <p>Selamat belajar HTML</p>
13  </dialog>
14
15  <script>
16    var tombol = document.getElementById('tombol');
17    var dialog = document.getElementById('dialog');
18
19    tombol.addEventListener('click', function() {
20      dialog.setAttribute("open", "open");
21    });
22  </script>
23
24 </body>
25 </html>
```



Gambar: Dialog baru tampil ketika tombol Tampilkan di klik

Disini saya membuat beberapa baris kode JavaScript agar dialog tampil ketika tombol **Tampilkan** di klik. Penjelasan kode JavaScript diatas tidak akan saya bahas karena akan menjadi jatah buku **JavaScript Uncover**.

17.7 Update HTML 5.3

Setelah HTML 5.1 dan HTML 5.2, W3C saat ini sudah langsung bekerja untuk membuat draft spesifikasi [HTML 5.3](#)⁶.

Dijadwalkan, **HTML 5.3** bisa rampung di akhir tahun 2018. Menarik untuk menunggu apa saja fitur-fitur HTML baru yang akan ditambahkan oleh W3C nantinya.

⁶<https://www.w3.org/TR/html53/changes.html#new-features>

Kursus Kilat JavaScript Dasar

JavaScript adalah bahasa pemrograman web yang digunakan untuk membuat interaksi (*behavior*) ke dalam halaman web. *JavaScript* merupakan bagian dari 3 teknologi penting yang menjadi dasar website modern, yakni **HTML** untuk membuat struktur konten (isi dari website), **CSS** untuk tampilan (presentation), dan **JavaScript** untuk interaksi (behavior).

Berbeda dengan HTML, **JavaScript** adalah bahasa pemrograman ‘murni’ yang berarti ia memiliki struktur pemrograman lengkap seperti *variabel*, *tipe data*, *perulangan*, *kondisi IF*, *function*, dan *objek*.

Dalam bab tambahan ini saya akan memberikan sedikit pengantar untuk memahami **JavaScript**. Pembahasan mengenai **JavaScript** sendiri sangat luas, dan saya menyarankan anda untuk mulai mempelajari **JavaScript** setelah membaca buku ini. **JavaScript** sangat powerful dan selalu dibutuhkan dalam pembuatan website modern.

Pembahasan **JavaScript** kali ini hanya sebagai pelengkap dan panduan sebagai dasar untuk fitur **HTML5** yang mengharuskan penggunaan **JavaScript**, seperti **canvas** dan **validasi form**.

Apa itu **JavaScript**?

JavaScript adalah bahasa pemrograman web berbasis client, yang artinya berjalan di web browser (berbeda dengan bahasa PHP yang berjalan di sisi server). **JavaScript** digunakan untuk menambahkan fitur interaksi pada halaman web.

Sebagai contoh, jika anda ingin teks tampil setelah men-klik tombol tertentu, membuat background berubah warna, atau membuat gambar yang tampil sebagai slideshow, semua ini hanya bisa dilakukan dengan bantuan **JavaScript**.

Saat ini **JavaScript** sudah menjadi fitur wajib website modern. Library **JavaScript** seperti [jQuery⁷](#), atau framework **JavaScript** seperti [Angularjs⁸](#) juga semakin populer digunakan.

Cara Menggunakan **JavaScript**

Sama seperti CSS, kita menyisipkan kode **JavaScript** ke dalam file **HTML**. Jika dilihat dari lokasi tempat kode program **JavaScript** berada, terdapat 3 cara menggunakan **JavaScript**:

- **Inline **JavaScript****
- **Internal **JavaScript****
- **External **JavaScript****

⁷jquery.com

⁸<https://angularjs.org>

Inline JavaScript

Metode inline JavaScript digunakan untuk menyisipkan kode JavaScript langsung ke dalam tag HTML. Ini dilakukan dengan menggunakan **event**.

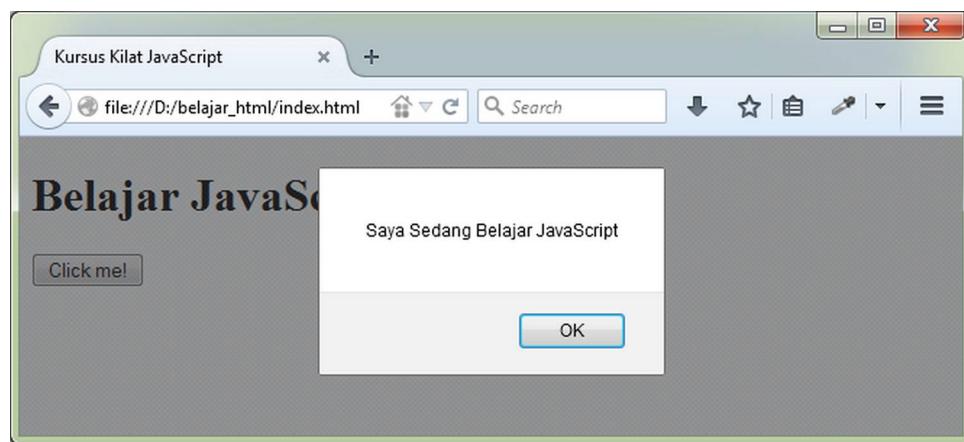
Event adalah suatu tindakan yang bisa dilakukan kepada sebuah objek HTML. Sebagai contoh, sebuah tombol bisa dikenakan event: *click*, *double click*, *right click*, *mouse over* (cursor mouse berada di atas tombol), dll. Terdapat belasan event yang bisa dilakukan kepada sebuah objek.

Untuk dapat menggunakan event ini dengan metode **inline JavaScript**, kita tinggal menambahkan kata “**on**”, sehingga untuk membuat interaksi ketika sebuah tombol di-klik, kita menggunakan atribut **onclick**.

Berikut contoh penggunaan Inline JavaScript:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Kursus Kilat JavaScript</title>
6 </head>
7 <body>
8   <h1>Belajar JavaScript</h1>
9   <button onclick="alert('Saya Sedang Belajar JavaScript')">
10  Click me!</button>
11 </html>
```



Gambar: Tampilan alert JavaScript

Jika anda menjalankan kode di atas, dan men-klik tombol ‘Click me!’, akan tampil kotak pesan ‘Saya Sedang Belajar JavaScript’. Ini dibuat menggunakan JavaScript. Bagian “`alert('Saya Sedang Belajar JavaScript')`” adalah kode JavaScript yang disisipkan ke dalam event **onclick**.

Cara penggunaan JavaScript dengan metode **inline JavaScript** seperti ini sangat praktis, namun tidak disarankan karena langsung bergabung dengan HTML.

Internal JavaScript

Jika menggunakan metode **Internal JavaScript**, kita bisa memindahkan kode JavaScript ke dalam tag `<script>`. Lokasi tag `<script>` bisa diletakkan dimana saja, apakah di bagian `<head>`, di awal `<body>`, atau di akhir halaman.

Berikut modifikasi kode program kita sebelumnya. Kali ini menggunakan metode Internal JavaScript:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Kursus Kilat JavaScript</title>
6   <script>
7     function pesan(){
8       alert('Saya Sedang Belajar JavaScript');
9     }
10    window.onload = function()
11    {
12      var a = document.getElementById("tombol");
13      a.addEventListener("click", pesan);
14    }
15  </script>
16 </head>
17 <body>
18   <h1>Belajar JavaScript</h1>
19   <button id="tombol"> Click me!</button>
20 </html>
```

Pada contoh di atas, kode JavaScript berada di bagian `<head>` dari halaman. Hal ini membuat halaman menjadi rapi karena seluruh kode JavaScript berada di bagian head.

Jika anda menjalankan halaman di atas, hasilnya masih sama dengan contoh sebelumnya. Tapi kali ini saya menggunakan beberapa kode tambahan.

Karena kita tidak menulis atribut “`onclick`” secara langsung ke dalam tag `<button>`, maka kita perlu mencari cara untuk ‘mengaitkan’ event `click` kepada tombol ‘Click me!’ tersebut.

Untuk mencari sebuah objek HTML dengan JavaScript, bisa menggunakan perintah `document.getElementById()`. Ini adalah sebuah *function* (lebih tepatnya: *method*) yang berfungsi untuk mencari objek HTML yang memiliki atribut `id`.

Dari kode HTML dapat dilihat bahwa tombol ‘Click me!’ memiliki atribut `id="tombol"`, maka saya bisa menggunakan perintah `document.getElementById("tombol")` untuk mendapatkan objek tersebut. Agar mudah diakses, saya menyimpan objek tombol ini ke dalam variabel `a`, sehingga penulisan kodennya menjadi:

```
var a = document.getElementById("tombol");
```

Selanjutnya, agar tombol ‘Click me!’ dapat berfungsi ketika di-click, kita menambahkan event “click” menggunakan perintah:

```
a.addEventListener("click", pesan);
```

addEventListener adalah method JavaScript yang digunakan untuk menambahkan sebuah event, dimana dalam contoh di atas adalah event **click**. Namun apa yang terjadi ketika tombol di klik? Inilah yang dijelaskan menggunakan fungsi pesan, dimana fungsi ini hanya untuk menampilkan kotak `alert('Saya Sedang Belajar JavaScript')`.

Huff... anda masih bersama saya?

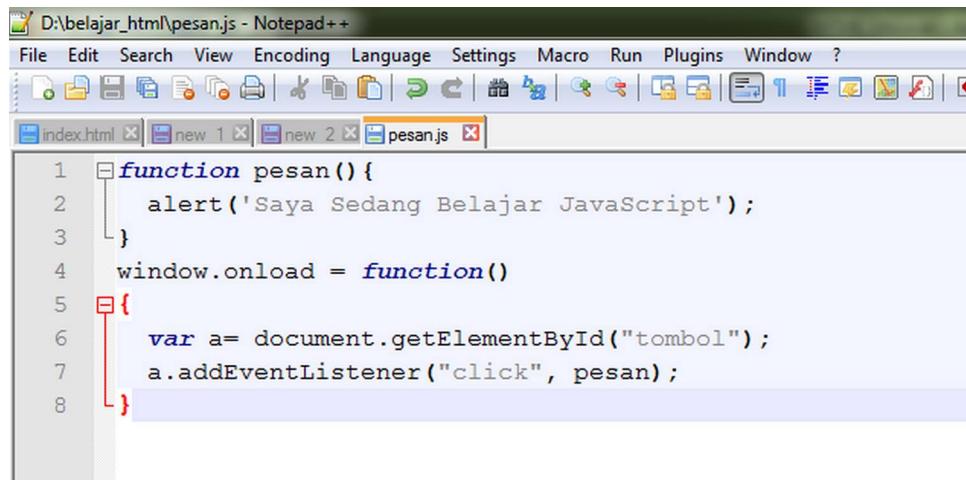
Kode-kode di atas mungkin terkesan rumit, terutama apabila anda belum pernah mencoba bahasa pemrograman komputer. Di dalam JavaScript, kita akan banyak berurusan dengan *variabel, function, method, object* yang memang lebih kompleks jika dibandingkan dengan HTML maupun CSS.

External JavaScript

Cara ketiga untuk memasukkan kode JavaScript adalah dengan metoda **External JavaScript**. Kali ini, kita memindahkan seluruh kode JavaScript ke dalam file external yang berekstensi *.js, kemudian diakses dari HTML.

Sebagai contoh, saya akan memindahkan seluruh kode JavaScript sebelumnya ke dalam file: **pesan.js**, yang isinya adalah sebagai berikut:

```
function pesan(){
    alert('Saya Sedang Belajar JavaScript');
}
window.onload = function()
{
    var a= document.getElementById("tombol");
    a.addEventListener("click", pesan);
}
```



```

1  function pesan(){
2      alert('Saya Sedang Belajar JavaScript');
3  }
4  window.onload = function()
5  {
6      var a= document.getElementById("tombol");
7      a.addEventListener("click", pesan);
8  }

```

Gambar: Cara membuat file pesan.js

Untuk halaman utama, berikut kode HTMLnya:

index.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="UTF-8">
5     <title>Kursus Kilat JavaScript</title>
6     <script src="pesan.js"></script>
7 </head>
8 <body>
9     <h1>Belajar JavaScript</h1>
10    <button id="tombol"> Click me!</button>
11 </html>

```

Dengan menggunakan external JavaScript, kode HTML menjadi ‘bersih’, dan file pesan.js juga bisa digunakan oleh lebih dari 1 halaman.

Menggunakan tag <noscript>

Salah satu kelemahan JavaScript (dan juga kelebihannya), adalah fitur JavaScript ini bisa dimatikan dari web browser. Beberapa web browser khusus juga ada yang tidak memiliki fitur JavaScript sama sekali.

Dalam situasi seperti ini kita bisa menampilkan pesan ke pengguna bahwa situs kita membutuhkan JavaScript untuk beroperasi. Caranya, dengan menggunakan tag <noscript>. Isi dari tag ini hanya akan ditampilkan pada web browser yang tidak memiliki JavaScript (atau sengaja dimatikan).

Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Belajar JavaScript</title>
5   <script src="pesan.js"></script>
6 </head>
7 <body>
8   <h1>Belajar JavaScript</h1>
9   <button id="tombol"> Click me!</button>
10  <noscript>Maaf, situs ini hanya bisa diakses dengan
11    JavaScript aktif</noscript>
12 </body>
13 </html>
```



Gambar: Teks pada tag <noscript> ditampilkan jika JavaScript tidak aktif

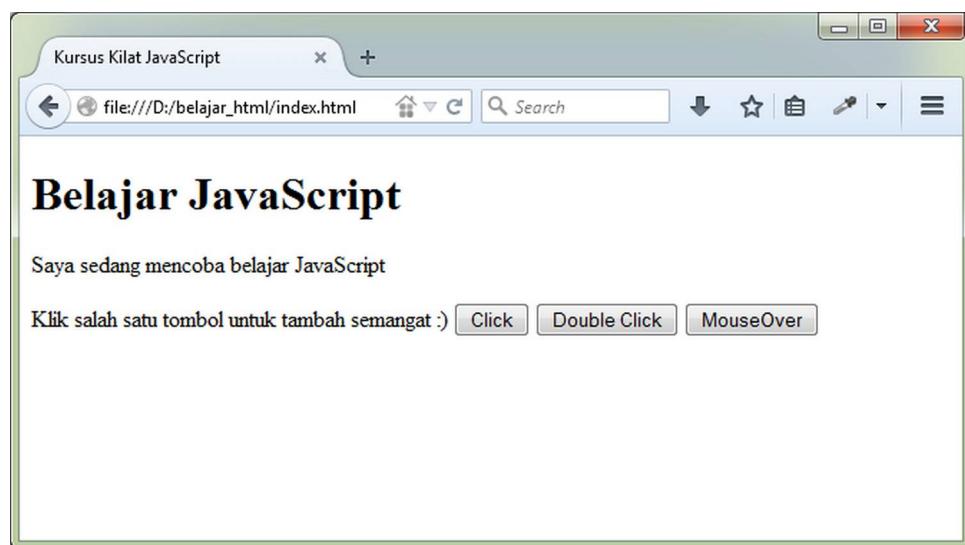
Teks ‘Maaf, situs ini hanya bisa diakses dengan JavaScript aktif’ hanya akan tampil ketika kode di atas dibuka dari web browser yang tidak mendukung JavaScript atau JavaScript sengaja tidak diaktifkan.

Sekilas Event JavaScript

Untuk contoh lain berkaitan dengan event dalam JavaScript, perhatikan kode berikut:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Kursus Kilat JavaScript</title>
5   <script type="text/javascript">
6     function tambah_semangat()
7     {
8       var a=document.getElementById("div_semangat");
9       a.innerHTML+="<p>Sedang Belajar JavaScript, Semangat...!!!</p>";
10    }
11   </script>
12 </head>
13 <body>
14   <h1>Belajar JavaScript</h1>
15   <p> Saya sedang mencoba belajar JavaScript</p>
16   Klik salah satu tombol untuk tambah semangat :)
17   <button id="tambah" onclick="tambah_semangat()">Click</button>
18   <button id="tambah" ondblclick="tambah_semangat()">Double Click</button>
19   <button id="tambah" onmouseover="tambah_semangat()">MouseOver</button>
20   <div id="div_semangat"></div>
21 </body>
22 </html>
```



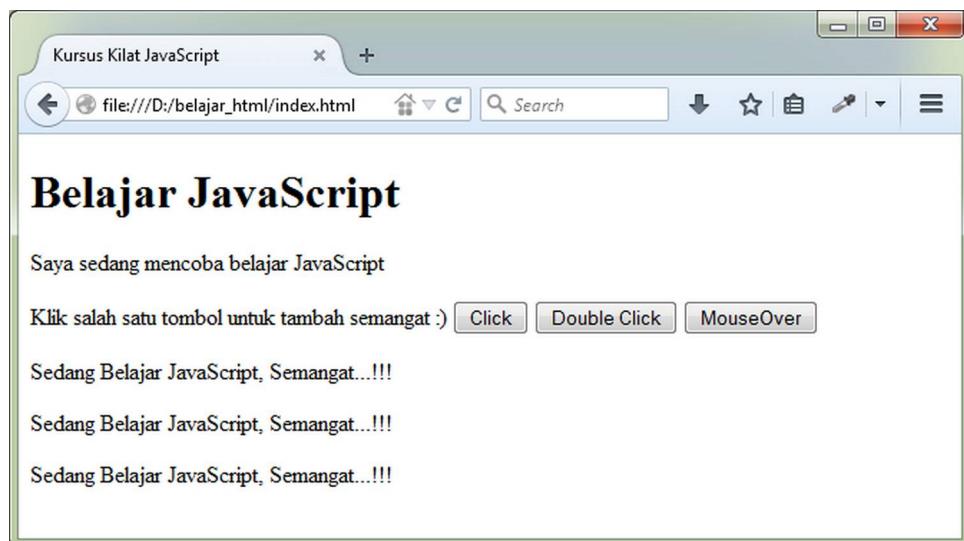
Gambar: Mencoba event 'onclick', 'ondblclick' dan 'mouseover'

Dalam kode di atas, saya menggunakan 3 event JavaScript:

- 'onclick': aktif ketika di klik.
- 'ondblclick': aktif ketika double klik.

- ‘mouseover’: aktif ketika mouse di atas tombol.

Silahkan dicoba untuk mengklik salah satu tombol, sesuai dengan event yang ada.



Gambar: Mencoba event ‘onclick’, ‘ondblclick’ dan ‘mouseover’

Dalam kursus kilat JavaScript ini kita hanya baru menyentuh sedikit saja dari kemampuan JavaScript. Sangat banyak aplikasi yang bisa dibuat menggunakan JavaScript. Terlebih lagi berbagai fitur HTML5 harus menggunakan JavaScript, seperti *canvas*, *validasi*, dan *HTML5 API*. Bahkan kita juga bisa membuat games menggunakan JavaScript.

Dalam konsep web modern, HTML, CSS dan JavaScript adalah 3 teknologi yang wajib dipelajari. Berbagai library JavaScript seperti [jQuery⁹](#) juga sangat memudahkan penggunaan JavaScript. Berani mencobanya? :)

⁹jquery.com

Kursus Kilat CSS Dasar

Walaupun buku ini fokus membahas HTML, tetapi banyak materi yang berkaitan dengan CSS. Oleh karena itu saya merasa perlu untuk membuat pengantar singkat tentang CSS.

Pembahasan tentang CSS ini hanya sekedar pengenalan. CSS adalah materi yang cukup luas dan saya sangat menyarankan anda untuk mulai mempelajari CSS setelah membaca buku ini. Jika anda tertarik dengan web design, maka HTML+CSS adalah pengetahuan yang wajib dikuasai.



Saat ini di DuniaIlkom sudah tersedia eBook [CSS Uncover¹⁰](#) yang akan membahas CSS dengan lebih detail. eBook ini merupakan lanjutan paling pas dari [HTML Uncover](#).

Apa itu CSS?

CSS adalah singkatan dari **Cascading Style Sheet**. Sebagaimana HTML, CSS bukanlah sebuah bahasa pemrograman, tetapi berupa kode-kode perintah sederhana dengan aturan penulisan tertentu.

CSS berfungsi untuk mengatur tampilan dari HTML. Jika HTML bertanggung jawab untuk membuat struktur halaman, maka CSS bertugas untuk ‘mewarnai’ halaman tersebut.

Cara Menggunakan CSS

Untuk menggunakan CSS, kita tinggal ‘menyisipkan’ kode tersebut ke dalam halaman HTML. Terdapat 3 metode untuk menginput kode CSS:

- **Inline CSS**

Metode inline CSS akan menyisipkan kode CSS langsung ke dalam tag HTML. Ini dilakukan menggunakan atribut **style**.

- **Internal CSS**

Metode internal CSS dilakukan dengan menginput kode CSS pada bagian `<head>` dari sebuah halaman HTML. Kita menggunakan tag `<style>` untuk keperluan ini.

- **External CSS**

Metode external CSS adalah menginput file CSS pada bagian `<head>` dengan menggunakan tag `<link>`. Dengan cara ini, sebuah file CSS bisa digunakan oleh banyak halaman HTML.

¹⁰<http://www.duniaIlkom.com/css-uncover-panduan-belajar-css-lengkap-untuk-pemula/>

Pengertian Selector, Property dan Value

Selector, Property dan Value merupakan struktur dasar dari CSS. Berikut adalah contoh penulisan kode CSS yang melibatkan ketiganya:

```
p {  
    font-size: 14px;  
    color: blue;  
}
```

Kode CSS diatas bertujuan untuk mengubah ukuran font menjadi 14 pixel dan berwarna biru untuk tag `<p>`.

Sebuah kode CSS diawali dengan **selector**. *Selector* berfungsi untuk ‘memilih’ tag atau bagian mana dari halaman HTML yang ingin diubah tampilannya (yang akan di-style). Pada contoh diatas ‘p’ berfungsi sebagai **selector**.

Dalam CSS terdapat banyak jenis-jenis selector. Selector paling sederhana adalah **tag selector** atau **element selector**, dimana kita menulis jenis tag apa yang akan di-style (tag ditulis tanpa kurung siku). Sehingga jika saya ingin men-style tag `<p>`, maka tinggal menulis: `p`, apabila ingin men-style tag `<h1>`, maka tinggal menulis: `h1`.

Setelah selector, penulisan berikutnya adalah **property** beserta **nilai-nya**. Di dalam CSS, **Property** adalah ‘sesuatu’ yang bisa diubah tampilannya, seperti jenis tulisan, besar huruf, warna huruf, lebar tabel, tinggi kolom, dll. Terdapat ratusan *property* di dalam CSS. Mayoritas property bersifat global (bisa digunakan pada selector apa saja), dan sebagian lagi hanya bisa digunakan untuk selector tertentu.

Dalam contoh diatas, **font-size** dan **color** adalah **property**. `font-size` digunakan untuk mengatur ukuran teks dan `color` digunakan untuk mengatur warna dari teks tersebut.

Untuk memberikan nilai kepada property tersebut, kita menggunakan tanda *titik dua* ‘ : ’ dan kemudian diikuti dengan nilainya. Pada contoh diatas, jika saya ingin mengubah ukuran teks menjadi 14 pixel, maka penulisannya menjadi: `font-size: 14px`.

Setiap pasangan property dan nilai CSS diakhiri dengan tanda titik koma ‘ ; ’, dan seluruh pendeklarasian **property** harus berada di dalam kurung kurawal: ‘ { ’ dan ‘ } ’.

Contoh penggunaan Internal CSS

Setelah memahami aturan dasar penulisan CSS, mari kita lihat bagaimana cara penggunaannya. Kita akan membahas terlebih dahulu cara menggunakan *metode internal CSS*, yakni menempatkan kode CSS pada bagian `<head>` dengan menggunakan tag `<style>`. Berikut contoh kode programnya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Kursus Kilat CSS</title>
6   <style>
7     h1 {
8       color:red;
9     }
10    p {
11      color:green;
12      font-size:16px;
13      font-family: Tahoma, Geneva, sans-serif;
14    }
15  </style>
16 </head>
17 <body>
18   <h1>Belajar HTML</h1>
19   <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit,
20   sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
21   Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris
22   nisi ut aliquip ex ea commodo consequat.</p>
23   <h1>Belajar CSS</h1>
24   <p>Duis aute irure dolor in reprehenderit in voluptate velit esse cillum
25   dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non
26   proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
27 </body>
28 </html>
```



Gambar: Tampilan halaman dengan internal style CSS

Halaman HTML diatas terdiri dari 2 buah tag `<h1>` dan 2 buah tag `<p>`. Saya menyisipkan kode CSS untuk mengubah warna dari tag `<h1>`, serta mengubah warna, jenis font dan ukuran font untuk tag `<p>`. Perhatikan bahwa saya menggunakan **tag selector**, sehingga seluruh tag `<h1>` dan tag `<p>` akan berubah tampilannya.

Sebagaimana yang telah dijelaskan, di dalam CSS terdapat banyak jenis selector. Selain **tag selector**, dalam contoh berikut ini saya menggunakan **class selector** dan **id selector**:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Kursus Kilat CSS</title>
6   <style>
7     #judul {
8       color:#39538C;
9     }
10    .pertama {
11      color:#727272;
12      font-size:1.4em;
13      font-family: "Courier New", Courier, monospace;
14    }
15  </style>
16 </head>
17 <body>
18   <h1 id="judul">Belajar HTML</h1>
19   <p class="pertama">Lorem ipsum dolor sit amet, consectetur adipisicing
20   elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
21   Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris
```

```
22 nisi ut aliquip ex ea commodo consequat.</p>
23 <h1>Belajar CSS</h1>
24 <p>Duis aute irure dolor in reprehenderit in voluptate velit esse cillum
25 dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non
26 proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
27 </body>
28 </html>
```

Kali ini saya menggunakan selector **#judul** dan **.pertama**.

Selector yang diawali dengan tanda pagar ‘#’ disebut dengan **id selector**. Selector ini akan mencari tag dengan atribut **id**, sehingga selector **#judul** akan cocok dengan tag apapun yang memiliki atribut **id="judul"**. Dalam contoh diatas, tag **<h1>** pertama memiliki atribut tersebut, sehingga akan di-style oleh CSS.

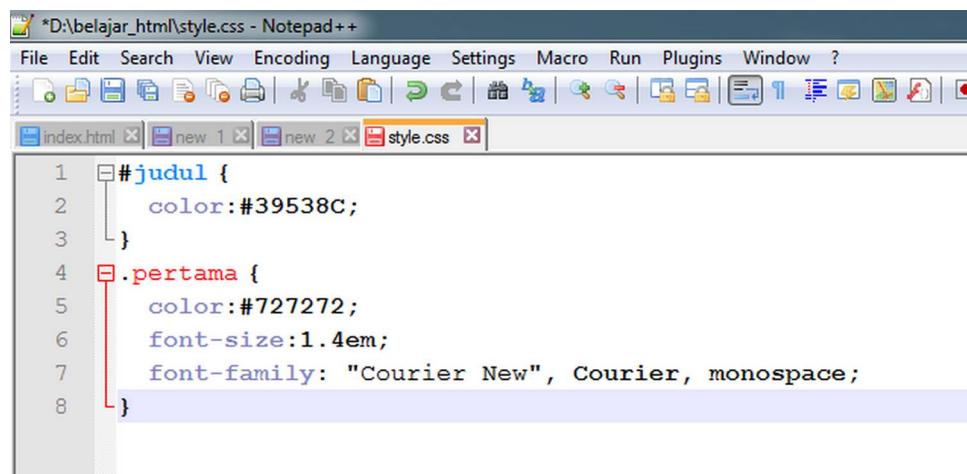
Selector yang diawali tanda titik ‘.’ disebut dengan **class selector**. *Class selector* akan mencari seluruh tag dengan atribut **class**, dimana selector **.pertama** akan men-style tag apapun yang memiliki atribut **class="pertama"**. Pada kode diatas, paragraf pertama memiliki atribut ini, sehingga tampilannya akan diubah sesuai dengan property CSS.

Contoh penggunaan External CSS

Metode Internal CSS sangat berguna jika kode CSS digunakan hanya untuk 1 halaman saja. Namun pada umumnya sebuah web terdiri dari beberapa bahkan ratusan halaman. Karena itu, akan jauh lebih efisien jika kode CSS tersebut ‘diangkat’ dari HTML dan ditempatkan pada file sebuah file khusus. File CSS ini kemudian di-link ke halaman HTML dengan menggunakan tag **<link>**.

Sebagai contoh, kita akan menggunakan folder **belajar_html**. Di dalam folder tersebut saat ini terdapat file **index.html** yang berisi kode HTML+CSS kita sebelumnya. Kali ini silahkan buat file teks baru dengan aplikasi Notepad++, klik menu **File-> New** kemudian ketikkan kode CSS berikut ini (atau cut-dan-paste dari halaman **index.html**):

```
#judul {
    color:#39538C;
}
.pertama {
    color:#727272;
    font-size:1.4em;
    font-family: "Courier New", Courier, monospace;
}
```



Gambar: Cara membuat file style.css

Save kode diatas pada folder belajar_html dengan nama **style.css**.

Selanjutnya, kembali ke halaman **index.html**, kita akan menambahkan tag `<link>` untuk menginput file CSS tadi ke dalam HTML, berikut kodennya:

index.html

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Kursus Kilat CSS</title>
6      <link href="style.css" type="text/css" rel="stylesheet">
7  </head>
8  <body>
9      <h1 id="judul">Belajar HTML</h1>
10     <p class="pertama">Lorem ipsum dolor sit amet, consectetur adipisicing
11         elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
12         Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris
13         nisi ut aliquip ex ea commodo consequat.</p>
14     <h1>Belajar CSS</h1>
15     <p>Duis aute irure dolor in reprehenderit in voluptate velit esse cillum
16         dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non
17         proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
18  </body>
19  </html>

```

Silahkan jalankan halaman **index.html**, dan tampilannya akan sama seperti halaman kita sebelumnya.

Contoh penggunaan Inline CSS

Cara terakhir untuk menginput kode CSS ke dalam HTML adalah dengan menggunakan metode **inline CSS**. Untuk menggunakan metode ini, kita langsung mengetik **property CSS** di dalam tag HTML dengan menggunakan atribut **style**. Berikut contoh penggunaannya:

index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Kursus Kilat CSS</title>
6 </head>
7 <body>
8   <h1 style="color:#39538C;">Belajar HTML</h1>
9   <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
10  eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim
11  ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut
12  aliquip ex ea commodo consequat.</p>
13  <h1>Belajar CSS</h1>
14  <p style="color:#727272;font-size:1.5em;">Duis aute irure dolor in
15  reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla
16  pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa
17  qui officia deserunt mollit anim id est laborum.</p>
18 </body>
19 </html>
```



Gambar: Tampilan halaman dengan inline style CSS

Perhatikan tag <h1> dan tag <p> pada baris ke 8 dan 14. Dalam kedua tag tersebut saya menggunakan atribut **style** yang didalamnya terdapat property CSS.

Penggunaan inline CSS seperti ini sebenarnya tidak disarankan, karena kita tetap menggabungkan HTML dan CSS, sehingga tidak ada bedanya dengan menggunakan atribut HTML reguler (dimana sebagian besar sudah dianggap **obsolete** atau **deprecated**).

Walaupun demikian, dalam contoh-contoh CSS pada buku ini saya akan banyak menggunakan inline CSS untuk menyederhanakan penulisan kode program. Untuk situs ‘live’ yang akan anda rancang sendiri, sebaiknya menggunakan metode external CSS.

Sekilas CSS Property

Seperti yang telah disebutkan, kekuatan CSS terletak dari kemampuan untuk ‘mencari’ tag HTML yang ingin diubah tampilannya (melalui selector), dan berbagai jenis style yang tersedia (melalui property).

Terdapat seratusan property yang ada di dalam CSS. Perkembangan CSS3 yang hampir bersamaan dengan kemunculan HTML5 juga membawa banyak property baru, seperti membuat bayangan, sudut bulat (*rounding corner*), hingga animasi yang sebelumnya hanya bisa dilakukan dengan JavaScript.

Sebagai contoh, berikut adalah beberapa property CSS yang bisa digunakan untuk mengubah tampilan teks. Kode-kode dibawah ini menggunakan inline CSS untuk menyederhanakan penulisan.

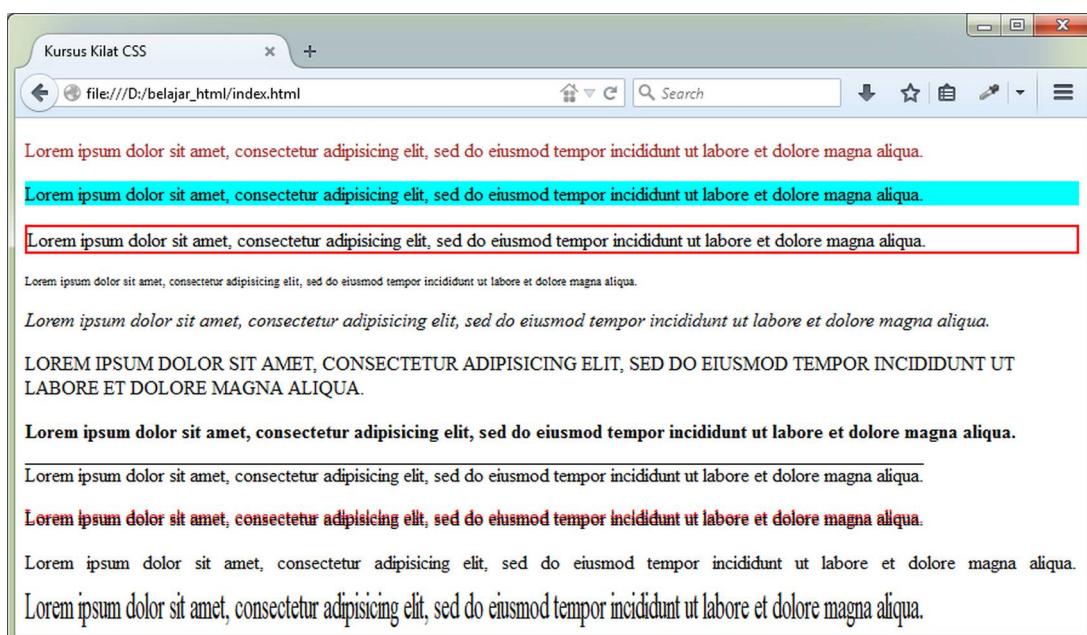
index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Kursus Kilat CSS</title>
6 </head>
7 <body>
8   <p style="color:brown;">
9     Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
10    eiusmod tempor incididunt ut labore et dolore magna aliqua.
11   </p>
12   <p style="background-color:aqua;">
13     Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
14    eiusmod tempor incididunt ut labore et dolore magna aliqua.
15   </p>
16   <p style="border: 2px solid red;">
17     Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
18    eiusmod tempor incididunt ut labore et dolore magna aliqua.
19   </p>
20   <p style="font-size:10px;">
21     Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
22    eiusmod tempor incididunt ut labore et dolore magna aliqua.
23   </p>
24   <p style="font-style:italic;">
25     Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
26    eiusmod tempor incididunt ut labore et dolore magna aliqua.
27   </p>
28   <p style="text-transform: uppercase;">
29     Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
30    eiusmod tempor incididunt ut labore et dolore magna aliqua.
31   </p>
32   <p style="font-weight:bold;">
33     Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
34    eiusmod tempor incididunt ut labore et dolore magna aliqua.
35   </p>
36   <p style="text-decoration: overline;">
37     Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
38    eiusmod tempor incididunt ut labore et dolore magna aliqua.
39   </p>
40   <p style="text-shadow: red 0 -2px;">
41     Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
42    eiusmod tempor incididunt ut labore et dolore magna aliqua.
43   </p>
44   <p style="word-spacing: 7px;">
```

```

45  Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
46  eiusmod tempor incididunt ut labore et dolore magna aliqua.
47  </p>
48  <p style="transform: scaleY(2);">
49  Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
50  eiusmod tempor incididunt ut labore et dolore magna aliqua.
51  </p>
52  </body>
53  </html>

```



Gambar: Tampilan berbagai property CSS untuk men-style teks

Seperti yang terlihat, sebuah teks yang sama bisa diubah tampilannya dengan mudah menggunakan CSS.

Seperti HTML, CSS cukup mudah dipelajari, namun sangat sulit untuk mahir :)

Saat ini tren perkembangan desain web juga telah beralih ke arah *mobile*. Membuat website yang bisa tampil bagus di layar besar seperti komputer / laptop sama pentingnya dengan di layar kecil seperti smartphone. CSS menjawab hal ini dengan konsep **Responsive Web Design**, dimana tampilan web akan ‘menyesuaikan’ diri dengan ukuran layar.

Framework CSS seperti [Bootstrap¹¹](#) dan [Zurb Foundation¹²](#) juga telah menjadi standar dalam membuat website yang responsive. Dan, semuanya diawali dengan pemahaman akan HTML dan CSS. Jadi, siap untuk belajar CSS?

¹¹getbootstrap.com

¹²foundation.zurb.com

Penutup: HTML Uncover

Sepanjang buku ini saya telah membahas ‘hampir’ seluruh aspek tentang HTML. Namun seperti yang anda lihat, kita sebenarnya masih berada di gerbang awal dunia web programming.

HTML merupakan pondasi dari seluruh teknologi pemrograman web. Pemahaman yang ‘solid’ tentang HTML akan membantu anda dalam mempelajari teknologi berikutnya.

Jadi, setelah ini saya harus mempelajari apa?

Jika anda berminat kepada web design, bisa lanjut ke CSS dan JavaScript. Jika anda ingin membuat website yang digabung dengan database, bisa ke PHP dan MySQL.

Namun sebagai ‘calon’ web programmer, atau ingin mempelajari proses pembuatan web secara utuh, saya anjurkan untuk mempelajari kelima materi dasar web programming: HTML, CSS, PHP, MySQL dan JavaScript.

Saat ini dunia web programming juga sudah begitu ‘rumit’, hampir mustahil kita bisa ahli di setiap bidang. Sebagai contoh, jika anda memilih mempelajari web design, tentu lanjut ke CSS.

Di dalam CSS nantinya juga terdapat teknologi lain yang dinamakan **CSS preprocessors**, seperti Sass atau Less. Kemudian juga terdapat **framework CSS** seperti Bootstrap atau Zurb Foundation. Semuanya digunakan untuk mendesain web.

Jika fokus ke web programming, anda bisa lanjut ke PHP. Di dalam PHP nantinya juga tidak kalah ‘njelime’. Selain harus memahami MySQL (untuk menyimpan data ke database), kita juga akan bersentuhan dengan **framework PHP** seperti Laravel, Code Igniter, Symfony, Yii Framework dll.

Help! pusing mas...

Dunia programming memang berkembang dengan sangat pesat. Bahkan dalam hitungan bulan akan selalu muncul ‘tools’ baru yang wajib dikuasai agar kita tidak ketinggalan.

Mudah-mudahan tutorial dan ebook DuniaIlkom akan selalu hadir untuk menyajikan bahan belajar yang berkualitas dan mudah dipahami. Atas doa dan dukungan donasi dari rekan-rekan pembaca yang membeli buku ini, saya akan terus berusaha menyajikan ebook berkualitas lainnya.

Sampai jumpa di ebook duniaIlkom selanjutnya :)



Update: Saat ini sudah tersedia eBook CSS, PHP, JavaScript dan MySQL Uncover DuniaIlkom.

Referensi HTML

Jika anda butuh informasi lengkap terkait tag, atribut dan element HTML, berikut beberapa sumber yang bisa menjadi referensi. Semua website ini berbahasa inggris, karena hingga saat ini belum ada situs berbahasa Indonesia yang lengkap dan update membahas seluruh tag-tag HTML.

MDN (Mozilla Developer Network) : <https://developer.mozilla.org>

Situs MDN bisa dibilang sejenis wikipedia-nya materi web programming. Berbagai programmer di seluruh dunia saling bekerjasama membuat sebuah referensi lengkap tentang HTML, CSS, dan JavaScript.

Khusus untuk referensi tag-tag HTML, bisa langsung kesini: [HTML element reference¹³](#).

HTML5Doctor: <http://html5doctor.com>

Website yang mengkhususkan diri membahas fitur-fitur terbaru HTML5, namun juga memiliki referensi lengkap tentang hampir semua element di dalam HTML: [HTML5 Element Index¹⁴](#).

Sitepoint: <https://www.sitepoint.com>

Sitepoint merupakan sebuah portal artikel dan materi seputar web programming (HTML, CSS, JavaScript, PHP dan MySQL). Artikel yang dibahas sebagian besar sudah masuk kategori advanced. Namun juga memiliki referensi untuk HTML: [HTML Reference¹⁵](#)

W3schools: <http://www.w3schools.com>

Situs yang satu ini dulunya banyak mendapat kritikan terkait pembahasan yang kurang update dan berbagai kesalahan. Bahkan muncul situs w3fools.com yang bertujuan mengkritik materi di w3schools. Nama situs ini juga mencoba ‘nompang’ ketenaran dari W3C (organisasi yang mengembangkan HTML).

Walaupun begitu, w3schools menyajikan tutorial dan referensi yang sangat mudah diikuti terutama bagi pemula. Materinya pun sekarang sudah diperbaiki sehingga cukup layak menjadi referensi. Untuk tutorial terkait HTML, bisa langsung ke: [HTML\(5\) Tutorial¹⁶](#).

¹³<https://developer.mozilla.org/en-US/docs/Web/HTML/Element>

¹⁴<http://html5doctor.com/element-index/>

¹⁵<http://reference.sitepoint.com/html>

¹⁶<http://www.w3schools.com/html/default.asp>

Daftar Pustaka

Sepanjang penulisan buku **HTML Uncover**, saya mengumpulkan bahan dari berbagai sumber. Anda bisa mengunjungi daftar pustaka ini untuk menambah pengetahuan seputar HTML dan HTML5.

Khusus untuk daftar pustaka yang berbentuk buku / eBook luar negeri, saya akses secara legal dengan berlangganan di safaribooksonline.com (sekitar US \$200 per tahun).

- Lloyd, Ian. **The Ultimate HTML Reference**. Sitepoint, 2008. ISBN: 9780980285888.
- **MDN (Mozilla Developer Network)**: <https://developer.mozilla.org/en-US/docs/Web/HTML>
- W3C: <https://www.w3.org/>
- **W3schools HTML Tutorial**: <https://www.w3schools.com/html/default.asp>
- **HTML5Doctor**: <http://html5doctor.com>
- **Sitepoint**: <https://www.sitepoint.com>