Daftar Isi

Daftar Gambar

Daftar Tabel

BAB 1
Internet dan Web

Internet telah mengubah pola aktivitas manusia baik pertemanan, belanja, pembayaran hingga pembelajaran. Perubahan ini didukung oleh semakin mudahnya akses ke internet. Profil internet Indonesia tahun 2022 yang dikeluarkan Asosiasi Penyelenggara Jasa Internet Indonesia (APJII) menyebutkan 77% penduduk di Indonesia telah menggunakan internet. Namun dari jumlah tersebut, masih sangat sedikit masyarakat yang memiliki *website* sendiri. Sebagai contoh, pelaku UMKM yang seharusnya membutuhkan *website* usaha, hanya 23,75% yang memiliki *website*.

Dalam pokok bahasan ini, Anda akan mempelajari pentingnya peran internet dan web, sejarah, dan istilah-istilah yang serupa namun tak sama.

Setelah menyelesaikan Bab ini, Anda diharapkan dapat:

1. Memahami asal muasal internet dan web
2. Memahami istilah-istilah internet dan web
3. Membedakan web statis dan dinamis

1.1. Pengantar Materi

Internet dan web saling mendukung satu sama lain. ARPAnet adalah internet versi pertama yang dibuat pada tahun

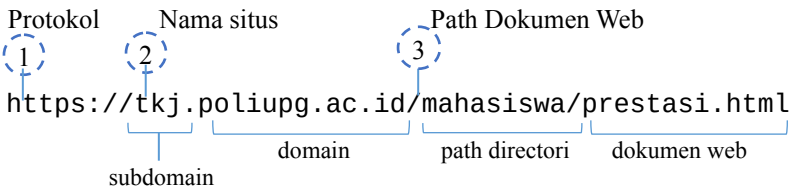
1969 yang berhasil mengirimkan paket data pertama kali melalui jaringan. Jaringan ini menghubungkan komputer yang ada di University of California, Los Angeles (UCLA) dan Universitas Stanford. Disusul oleh penemuan protokol komunikasi paket melalui jaringan pada tahun 1974. Protokol ini yang menjadi cikal bakal protokol TCP/IP. Pada masa tersebut, istilah *interconnected network* atau internet mulai digunakan. Pembangunan perangkat untuk *backbone* internet (1986) dan *internet service provider* (ISP) sebagai *internet exchange point* (1989) membentuk infrastruktur internet seperti yang ada sekarang. Seiring dengan perkembangan infrastruktur internet, layanan yang berjalan di atasnya juga mengalami transformasi terutama setelah teknologi *world wide web* (www) sudah dapat digunakan pada tahun 1991. WWW menjadi tonggak awal perkembangan web di dunia dan masih digunakan hingga saat ini.

1.2. Isi Materi

Konsep jaringan global yang ditawarkan internet sangat mendukung jangkauan dan kecepatan aktivitas manusia. Internet merupakan setumpuk perangkat keras yang terhubung sebagai jaringan besar dan di atasnya bekerja beberapa layanan. Layanan yang dimaksud antara lain email, web, transfer *file*, dan lain-lain. Sementara untuk melewatkan layanan ini, digunakan *protocol* sebagai jalur untuk melewatkan data layanan tersebut, misalnya untuk layanan web digunakan *hypertext transfer*

protocol (http) maupun *hypertext transfer protocol secure* (https) dan layanan email melalui protokol pop3.

Layanan web bukan internet. Jika internet merujuk pada keterhubungan jaringan dari seluruh dunia melalui *backbone* telekomunikasi dan *routing* menggunakan alamat IP, maka *world wide web* atau dikenal umum sebagai web merupakan layanan yang memungkinkan keterhubungan dokumen *hypertext* yang memanfaatkan internet sebagai media penghubungnya. Kita dapat melihat isi halaman web yang berisi teks, gambar, video, atau media lainnya serta dapat terhubung satu dengan yang lain melalui *hyperlink*. Alamat web lengkap untuk mengakses lokasi sumber daya pada web disebut *Unified Resources Locator* (URL).



Pola ini menunjukkan protokol komunikasi yang digunakan adalah https dengan nama situs tkj.poliupg.ac.id. Sebuah situs memiliki alamat IP publik, misalnya 159.223.81.66. IP publik merupakan alamat yang dapat dikunjungi dalam jaringan internet. Sementara alamat domain/subdomain adalah alias dari alamat IP yang sebenarnya. Penamaan alias ini diperlukan untuk memudahkan pengguna

mengingat alamat *website*. Perubahan nama alias menjadi alamat IP diatur oleh *Domain Name Server* (DNS). Tanpa DNS, kita harus mengetikkan alamat IP secara langsung untuk mengunjungi sebuah *website*.

1.2.1. World Wide Web (WWW)

Layanan web dimulai dari dibuatnya bahasa markup berisi *hypertext* oleh Tim Berners Lee di awal tahun 90-an. Bahasa ini diberi nama Hypertext Markup Language yang sangat populer dengan nama HTML. WWW merupakan kumpulan halaman-halaman HTML yang saling terkoneksi melalui *hyperlink* dan membentuk samudera informasi. WWW berjalan di atas protokol *hypertext transfer protocol (http).* WWW merupakan layanan internet yang paling popular hingga saat ini. Bahkan internet dapat digunakan secara luas setelah adanya layanan www.

Mosaic adalah web browser pertama yang digunakan khalayak umum pada tahun 1993. Browser buatan Andressen dan Eric Bina, yang merupakan mahasiswa Universitas Illonois pada saat itu, telah mampu membaca teks dan grafis. Browser dibutuhkan untuk menginterpretasikan dokumen web berisi sintaks HTML menjadi tampilan halaman web.

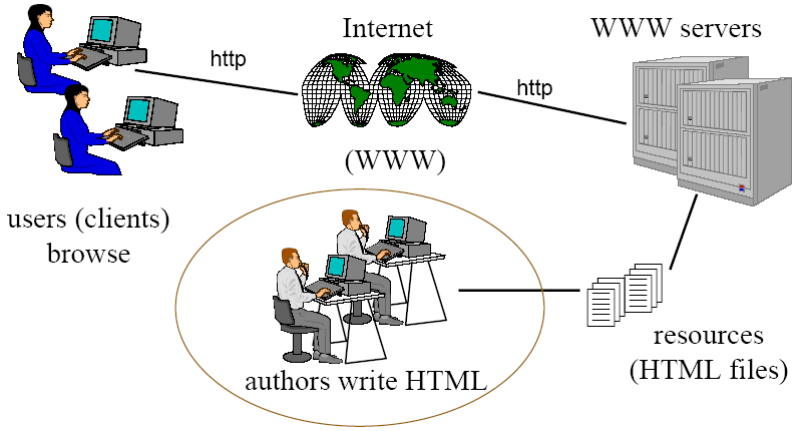
User/pengguna bertindak sebagai klien menggunakan browser untuk melakukan permintaan ke server web melalui internet. Di internet, pertama-tama nama *website*/ URL yang diklik akan dikirim ke DNS untuk mendapatkan alamat IP server

website yang dituju. Server ini harus selalu hidup untuk merespons permintaan dari pengguna.

1.2.2. Web Statis dan Dinamis

Dokumen web ditulis dengan HTML. Seorang web developer dapat menulis dokumen HTML secara langsung maupun terprogram menggunakan bahasa pemrograman tertentu. Berdasarkan hal tersebut, web dapat terbagi menjadi:

1. Web Statis, memiliki ciri-ciri isi halaman tidak pernah berubah. Dokumen web dapat saja diubah dengan cara mengubah dokumen HTML secara langsung. Biasanya digunakan untuk halaman Kontak Kami atau Tentang Kami. Gambaran cara kerja web statis dapat dilihat pada gambar 1.1. Semua isi dari halaman web terdiri dari dokumen HTML yang ditulis oleh seorang web developer. Ketika diakses, semua *file* yang diminta akan diunduh oleh browser dan dijalankan di sisi *client* (*client side scripting*).



Gambar 1. 1 Cara Kerja Web Statis

2. Web Dinamis, membutuhkan pemrograman web, sehingga isi halaman dapat diprogram sesuai kebutuhan. Isi web akan berubah tanpa melakukan perubahan pada dokumen HTML. Beberapa bahasa pemrograman web yang populer adalah PHP dan Javascript. Pemrograman yang memungkinkan terjadi perubahan dari sebuah halaman web secara dinamis. Misalnya tanggal yang menyesuaikan dengan waktu server, berita yang selalu berubah setiap waktu, hingga tampilan yang berbeda-beda di tiap dashboard pengguna. Semuanya diproses di server terlebih dahulu, kemudian server menjawab request dengan mengirimkan *resource* ke *client*. Web menjadi dinamis berkat *script* yang berjalan di sisi server (*server side scripting*)

1.3. Rangkuman

1. Internet adalah jaringan global berupa infrastruktur yang berupa *hardware*, sedangkan web adalah layanan yang memungkinkan untuk dilewatkan melalui internet.
2. Internet berkembang pesat dengan kemunculan www yang memungkinkan interkoneksi data secara luas
3. Pembuatan web membutuhkan kemampuan desain dan pemrograman, yaitu membuat tampilan serta menggunakan program untuk menciptakan dinamika sebuah web.

1.4. Tugas Latihan

1. Jelaskan pengertian dari:
   a. Social media
   b. E-commerce
   c. Internet
   d. Web
   e. Domain Name Server (DNS)
2. Jelaskan perbedaan
   a. Unified Resource Locator (URL) dan Unified Resource Identifier (URI)
   b. Web dan *website*
   c. Web Statis dan dinamis
   d. Social media dan *Instant Messenger*
3. Jelaskan cara kerja web statis!

4. Apa fungsi pemrograman dalam membangun sebuah web dinamis?
5. Sebutkan bahasa pemrograman yang dapat digunakan untuk membangun web dinamis

1.5. Pustaka

1. Brpadbamdsearch.com, [Online] tersedia di https://www.broadbandsearch.net/blog/who-invented-the-internet-full-history, diakses tanggal 16 Juni 2022
2. Ducket, J., HTML & CSS Design and Build *Website*s, John Wiley & Sons, 2011
3. Gil, Paul, [Online] tersedia di https://www.lifewire.com/top-internet-terms-for-beginners-2483381, diakses tanggal 16 Juni 2022

BAB 2
Desain Web

Dalam pokok bahasan ini, Anda akan mempelajari pentingnya peran desain web, memahami kebutuhan pengguna, dan tahapan-tahapan desain.

Setelah menyelesaikan Bab ini, Anda diharapkan dapat:

1. Memahami peran desain web
2. Memahami kebutuhan desain bagi pengguna
3. Menjelaskan tahapan desain

2.1. Pengantar Materi

Membangun sebuah *website* bukan hanya membutuhkan keterampilan bahasa pemrograman. Namun, pengetahuan tentang target pengguna *website* adalah kunci kesuksesan sebuah *website*. Setelah memahami target pengguna barulah tampilan web dirancang. Untuk menjawab kebutuhan pengguna (*user*) terhadap *website* dibutuhkan seorang *User Interface* dan *User Experience* (UI/ UX) *designer*.

2.2. Isi Materi

2.2.1. Kebutuhan Pengguna

Pengunjung *website* adalah pengguna utama *website* kita, sehingga kita perlu melihat dari perspektif pengunjung, antara lain:

A. Pengunjung tidak membaca keseluruhan halaman

Meskipun *website* penuh dengan informasi, pengunjung hanya akan menghabiskan beberapa detik untuk melihat-lihat. Penting untuk meletakkan inti dari halaman di kalimat awal, buat mereka melihat paragraf pendek dan petunjuk-petunjuk pada halaman. Gunakan gambar atau objek dinamis untuk menonjolkan petunjuk-petunjuk tersebut

B. Lebih Sedikit Lebih Baik

Buat paragraf, halaman, dan bagian-bagian lainnya dalam halaman sependek mungkin, agar halaman tidak dipenuhi dengan teks yang dapat membuat pengunjung jenuh. Jika memang banyak hal yang ingin disampaikan, tampilkan judul dan atau potongan bagian saja, sementara sisanya diletakkan di halaman yang berbeda.

C. Navigasi

Buat navigasi yang konsisten pada semua halaman di dalam situs. Hindari buat *hyperlink* di dalam paragraf yang mengarahkan pengunjung ke tiap halaman dalam situs. Ini akan mengacaukan konsistensi dalam struktur navigasi situs. Jika ingin menggunakan *hyperlink*, tambahkan di bagian akhir paragraf atau di bagian menu.

D. Kecepatan akses

Menurut penelitian yang dilakukan google, 53 persen pengunjung akan meninggalkan *website* Anda jika loadingnya lebih dari tiga detik. Dampaknya 79%

pengunjung tidak ingin kembali lagi, 49% di antara mereka kehilangan kepercayaan pada bisnis Anda, 44% di antaranya memberitahukan kekecewaan mereka ke orang lain. Jika proses *download* lambat, sebaiknya tampilkan grafis dengan ukuran *file* kecil.

E. Biarkan pengunjung berbicara

Pengunjung situs adalah "pelanggan" kita. Mereka bisa memberikan informasi-informasi untuk perbaikan kita. Mudahkan pengunjung menghubungi anda agar banyak masukan yang kita dapatkan dari berbagai aspek.

F. Perhatikan monitor pengguna

Tidak semua orang punya monitor yang sama dengan komputer Anda. Pastikan melakukan tes di ukuran monitor target pengunjung anda. Jika perlu, gunakan teknologi yang responsif dengan ukuran monitor.

G. Perhatikan browser yang digunakan

Untuk menguji coba situs, gunakan beberapa browser yang berbeda. Hal ini penting agar situs anda ditampilkan sebagaimana mestinya. Perhatikan perkembangan browser yang banyak digunakan pengunjung

H. *Plug-in* apa yang dimiliki pengguna

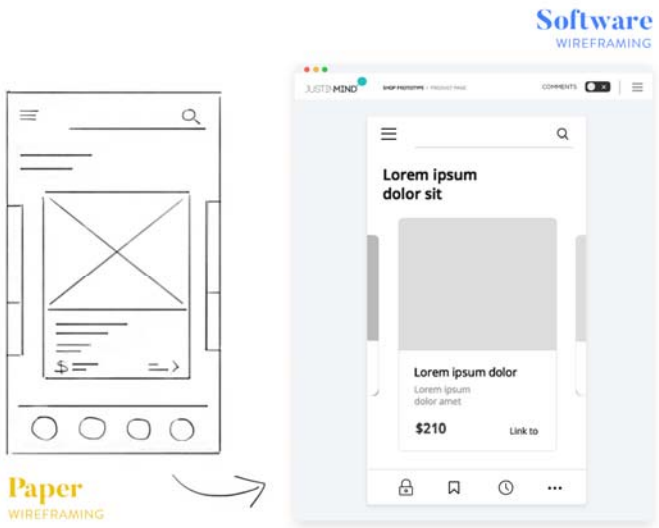
Format audio dan video biasanya memerlukan program terpisah (*plug-in*) untuk bisa diputar di halaman *website*. Pastikan pengunjung memiliki akses ke *plug-in* untuk menampilkan konten dalam situs kita.

I. Manfaatkan *search engine*

Beberapa *search engine* popular dapat digunakan sebagai media untuk memperkenalkan situs.

2.2.2. Desain *Wireframe, Mockup,* dan *Prototype*

Wireframe adalah kerangka desain paling dasar sebuah halaman web. Struktur layout berupa kerangka berbentuk tabel.



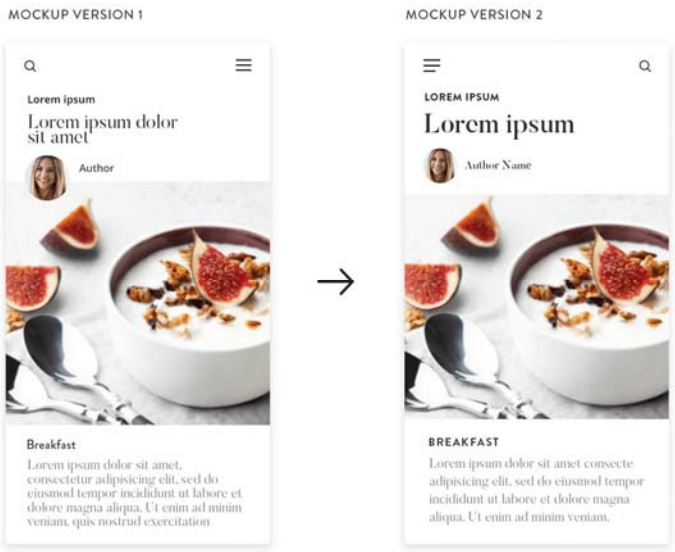
(a) Menggunakan Kertas (b) Menggunakan Aplikasi

Gambar 2. 1 Wireframe Sederhana
(Adiseshiah, 2016)

Fungsinya untuk mendapatkan umpan balik dari *user* sebelum fokus ke detail visualisasi. *Wireframe* belum menunjukkan halaman web yang sebenarnya.

Sementara *mockup* sudah memperlihatkan wujud grafis halaman *website*, hanya saja belum dapat berinteraksi langsung

dengan pengguna. Sementara *prototype* sudah dapat dikatakan halaman *website* yang sudah siap diakses hanya saja belum selengkap versi final. Perbedaan *wireframe* dan *mockup* dapat dilihat pada gambar 2.1 dan 2.2. Banyak aplikasi *online* (daring) yang yang tersedia untuk membuat *wireframe*, *mockup* bahkan *prototype*, misalnya Figma (figma.com)



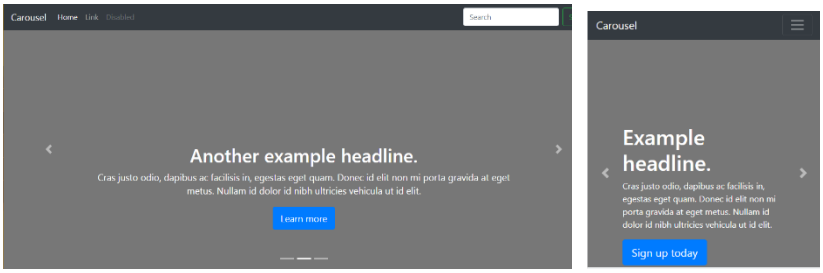
Gambar 2. 2 *Mockup* Halaman Web
(Adiseshiah, 2016)

2.2.3. Prototype

Untuk membuat tampilan halaman yang interaktif membutuhkan tiga layer utama, yaitu *content layer* (*file* .html), *presentation layer* (*file* .css), dan *behaviour layer* (*file* .js). HTML membentuk konten halaman, membangun struktur halaman dan menambahkan makna semantik. Halaman HTML

berfungsi sebagai penghubung *file* CSS and Javascript. CSS memberikan aturan pada halaman HTML yang menentukan bagaimana konten HTML ditampilkan antara lain *background, border, box dimension, color, font, etc*. Javascript mengatur cara halaman bertingkah laku, meningkatkan interaksi dengan pengguna.

Terdapat *framework* desain yang dibuat untuk menggabungkan tiga layer ini. Yang paling populer adalah bootstrap. Framework bootstrap sangat mudah digunakan, sehingga banyak template yang dibuat berbasis bootstrap. Selain itu bootstrap juga bersifat responsif, artinya tampilan dapat menyesuaikan dengan ukuran perangkat yang digunakan pengguna, sehingga dipastikan bisa bekerja secara optimal di semua ukuran layar. Bootstrap juga dapat digunakan semua browser. Kekurangan bootstrap adalah *resource* yang digunakan cukup besar baik *resource* css, js, glyph-icon, serta jQuery. Namun, tidak semua resource tersebut yang terpakai. Contoh tampilan web menggunakan boostrap seperti pada gambar 2.3.



(a) Tampilan Desktop (b) Tampilan mobile

Gambar 2. 3 Tampilan Responsif Bootstrap

2.3. Rangkuman

1. Desain web harus dibuat berdasarkan kebutuhan pengguna
2. Gambaran desain dapat dimulai dari *wireframe*, *mockup* hingga membuat *prototype*
3. Desain *prototype* menggunakan tiga layer yaitu HTML, CSS, dan Javascript.

2.4. Tugas Latihan

1. Buatlah desain *wireframe website* sekolah secara langsung berhadapan dengan pihak sekolah anda. Mintalah umpan balik untuk perbaikan desain anda.
2. Buatlah desain *mockup* halaman depan *website* tersebut menggunakan *tools* yang tersedia
3. Pastikan projek anda tersimpan untuk digunakan pada pembuatan *prototype* di materi selanjutnya.
4. Bagaimana tiga layer desain terhubung satu sama lain?
5. Apa saja kelebihan dan kekurangan menggunakan *framework* desain?

2.5. Pustaka

1. Boulton, M., Designing for The Web, [Online] tersedia di https://designingfortheweb.co.uk/ [Diakses tanggal 16 Juni 2022]
2. Adiseshiah, E. G., 2016. Wireframes vs. Mockups: what's

the best option?. [Online] tersedia di https://www.justinmind.com/blog/wireframes-and-mockups-whats-the-best-option/ [Diakses tanggal 16 Juni 2022].

3. Nayoan, A., 2021. 8+ Cara Mempercepat Loading *Website* dan Blog Anda [Online], tersedia di https://www.niagahoster.co.id/blog/cara-mempercepat-loading-blog [Diakses tanggal 16 Juni 2022]
4. Bootstrap, Example – Bootstrap [Online] tersedia di https://getbootstrap.com/docs/4.0/examples/ [Diakses tanggal 16 Juni 2022]

BAB 3
HTML(Hypertext Markup Language)

Dalam pokok bahasan ini, Anda akan mempelajari bentuk dasar dari HTML dan penggunaan dari beberapa tag dasar yang sering digunakan dalam pengembangan aplikasi web.

Setelah menyelesaikan bab ini, Anda diharapkan dapat:

1. Memahami bentuk dasar dari halaman web HTML
2. Menggunakan tag HTML dalam membuat halaman web
3. Membuat halaman HTML dengan komponen yang sesuai dengan kebutuhan

3.1. Pengantar Materi

Hypertext Markup Language atau HTML adalah bahasa markup yang digunakan untuk membuat sebuah halaman web. HTML5 adalah standar HTML terbaru yang digunakan untuk menulis sebuah dokumen web. Sebelumnya, standar yang digunakan adalah HTML 4 yang memiliki standar penulisan yang identik dengan XHTML yang lebih rapi dan ketat. XHTML dibuat untuk memudahkan *web developer* beralih dari HTML ke XML.

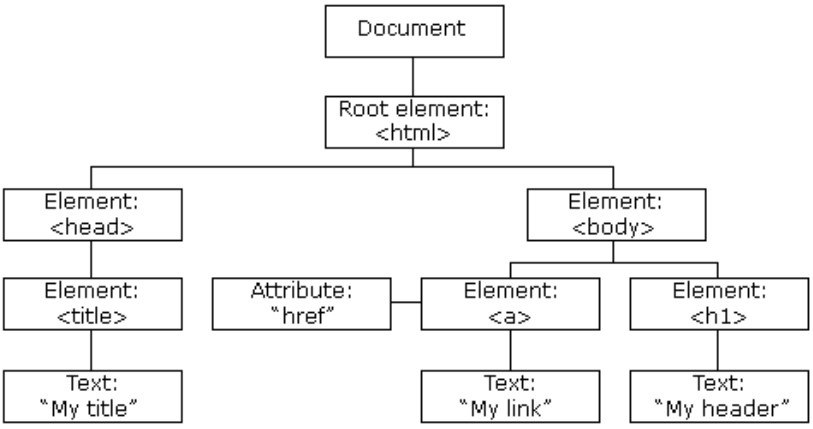
Namun dalam perjalanannya, HTML5 lebih mendominasi dengan mendapatkan dukungan semua browser, sehingga HTML5 lebih banyak digunakan hingga saat ini. Kelebihan lain HTML5 adalah:

- Memiliki elemen-elemen modern antara lain elemen video, audio dan *canvas* yang membuat halaman web semakin terlihat modern dengan mendukung multimedia
- Tag penutup dapat dihilangkan untuk *tag* kosong
- Adanya tag semantik, seperti *header, footer, section, article,* dan *nav* yang dapat mengurangi penggunaan *tag*
- Tidak *case-sensitive*
- Dukungan API yang memungkinkan pengguna berbagi lokasi

3.2. Isi Materi

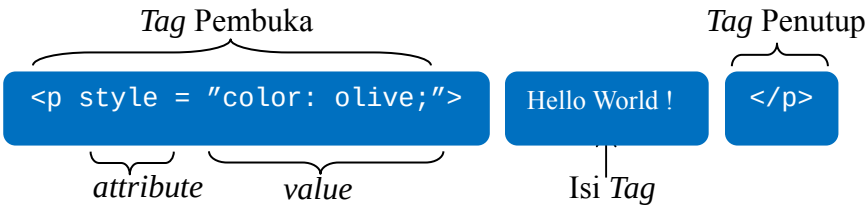
3.2.1 Struktur HTML

Sebuah dokumen web diinterpretasi oleh browser berdasarkan informasi dari struktur HTML. Struktur HTML pada gambar 3.1 menunjukkan elemen pada HTML bersifat bersarang.



Gambar 3. 1 Struktur Dokumen HTML

Sebuah elemen HTML ditulis dengan simbol *tag*. *Tag* yang kosong atau *empty tag* adalah *tag* yang tidak memiliki isi tag. Misalnya *tag breaking space*
.



Penjelasan tag di atas adalah sebagai berikut:

1. ***Tag*** merupakan simbol/tanda pembuka dan atau penutup dari sebuah elemen HTML.
2. **Atribut** merupakan ciri-ciri *Tag* yang dituliskan pada *Tag* pembuka.
3. **Isi *Tag*** merupakan isi dari *Tag* HTML biasanya berupa teks
4. **Elemen** merupakan komponen utama HTML yang mencakup *tag*, atribut, serta isi *tag*.

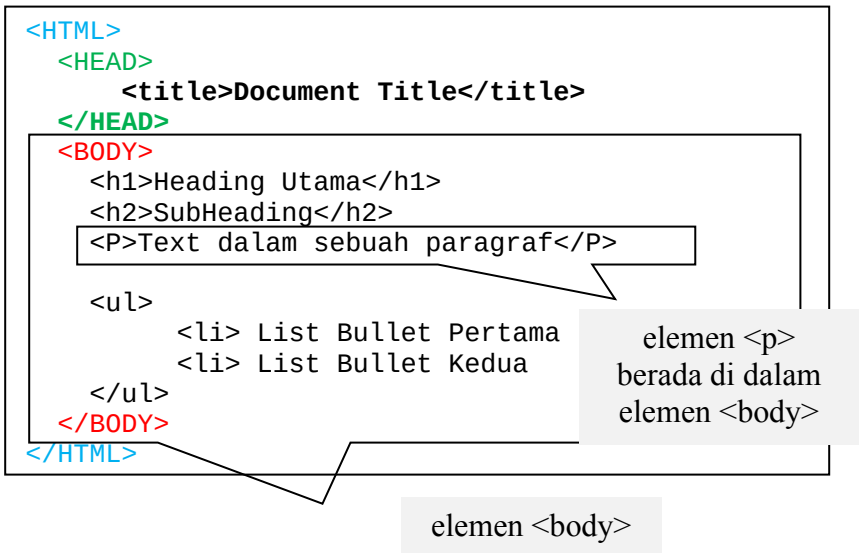
Berdasarkan pemahaman bentuk perintah HTML diatas, Anda akan belajar komponen-komponen HTML dasar selanjutnya. Penulisan *tag* di dalam dokumen HTML dapat dilihat pada gambar 3.2.

3.2.1 Elemen HTML

Sebuah elemen HTML yang memiliki isi harus memiliki *tag* pembuka dan penutup. Dan sebuah *tag* HTML selalu diapit dengan tanda lebih kecil < dan lebih besar >, misalnya:

```
<p> Contoh tag yang memiliki isi </p>
```

Selain itu, ada pula *tag* yang tidak memiliki isi, yang disebut *tag* kosong (*empty tag*), misalnya untuk memberi spasi baru: `<br>.`



Gambar 3. 2 Dokumen HTML

Ada pula tag HTML yang berbeda, tetapi hasilnya pada tampilan sama, contohnya:

a. *Bold* `<b>` dan *strong* `<strong>`.

   `<b>` memberi efek cetak tebal pada teks, `<strong>` pun demikian, tetapi *tag strong* memiliki makna semantik bahwa teks tersebut lebih penting dibanding teks di sekitarnya.

b. *italic* <i> dan *Emphasis* , *cite* <cite>,
 dan <cite> juga memberi efek cetak miring, tetapi keduanya memiliki makna semantik yang berbeda.

Berdasarkan fungsinya, *tag* HTML dapat dikategorikan seperti tabel 3.1.

Tabel 3. 1 Elemen HTML

Tag	Fungsi
STRUKTUR DASAR	
<!DOCTYPE>	Tipe dokumen
<html>	Format dokumen adalah HTML
<body>	Body (isi) sebuah halaman
<!--...-->	Komentar
TEKS	
*<h1> to <h6>	Heading (judul)
<p>	Paragraph (paragraf baru)
 	Breaking space (baris baru)
<hr />	Horizontal line (garis horizontal)
*<acronym>	Acronym (singkatan) (HTML4)
*<abbr>	Abbreviation (singkatan) (HTML5)
*<address>	Address, format teks untuk alamat
<b>	Bold (cetak tebal)
<bdo>	Bi-directional Override (arah penulisan), misalnya rtl
<big>	Big text
*<blockquote>	Long quotation
**	Centered text
*<cite>	Citation, format teks untuk alamat
*<code>	Code, format teks untuk kode program

*<del>	Deleted text, teks yang seolah-olah terhapus
*<dfn>	Definition term
*	Emphasized, format teks untuk penekanan
**<font>	Font
<i>	Italic
*<ins>	Inserted text, teks yang baru ditambahkan
<kbd>	Keyboard text
<pre>	Preformatted text
<q>	Short quotation
**<s>	Strikethrough, format teks tercoret
*<samp>	Sample computer code
<small>	Smaller text (ukuran teks diperkecil)
**<strike>	Strikethrough (coret)
*	Strong (teks dipertebal)
<sub>	Sub*script*ed (subskrip)
<sup>	Super*script*ed (superskrip)
*<tt>	Teletype text
**<u>	Underlined text
<var>	Variable text
**<xmp>	Preformatted text
Block	
*<section>	Grup bagian tertentu sebuah halaman
	Division (section) di dalam document
	Section dalam document
Form dan Label	
<form>	Form untuk menerima input dari *user*
<input />	Berbagai macam Inputan pada formular
<textarea>	Multi-line text

<button>	Button
<select>	Seleksi (drop-down)
<optgroup>	Grup opsi pada daftar seleksi
<option>	Opsi pada daftar select
<label>	Label untuk elemen dalam formular
<fieldset>	Border di sekitar elemen formular
<legend>	Caption untuk elemen fieldset
**<isindex>	Kata kunci pencarian dalam dokumen HTML
List	
	Unordered list
	Ordered list
	Butir pada List
**<dir>	Directory list
*<dl>	Definition list
*<dt>	Butir definition dalam definition list
*<dd>	Des*cript*ion sebuah butir pada definition list
**<menu>	Menu list
Table	
<table>	Table
<caption>	Table caption (judul table)
<th>	Table header (judul kolom)
<tr>	Table row (baris table)
<td>	Table data (data tabel)
<thead>	Grup Header (grup judul)
<tbody>	Grup Body (grup isi)
<tfoot>	Grup footer tabel (grup judul kaki)
<col />	Nilai Atribut untuk satu atau lebih kolom
<colgroup>	Grup kolom untuk mengatur format teks
Image	

*<figure>	Grup gambar dan keterangannya
<img />	Gambar
*<figcaption>	Keterangan gambar
<map>	Gambar berupa peta
<area />	Area di dalam gambar peta
Links	
<a>	Anchor (link jangkar)
<link />	Link dengan *file* external tipe tertentu
Frames	
**<frameset>	Frame set
**<frame />	Window (frame) dalam sebuah frameset
**<noframes>	Alternatif jika frames tidak didukung
<iframe>	Inline frame
Style	
<*style*>	*style* info di dalam dokumen HTML
Meta Info	
<head>	Informasi tentang document
<title>	Title (judul) document
<meta>	Metadata dari document HTML
<base />	Default URL path untuk semua resource
**<basefont />	Default font untuk text dalam halaman
Programming	
<*script*>	Skrip, yang paling populer adalah Javascript
<no*script*>	Skrip jalan jika format *script* tidak didukung
**<applet>	Embedded applet
<object>	Embedded object
<param />	Parameter untuk elemen object

***Ket**: *Tag semantik., **Tag usang di HTML5*

Beberapa tag usang pada HTML4 memiliki padanan pada HTML5 seperti pada tabel 3.2

Tabel 3. 2 Tag Usang HTML4 dan Padanannya di HTML5

HTML4	Padanan di HTML5
<applet>	<objek>
<basefont>	<body *style*="font-family:Arial;
<blockquote>	<div *style*="margin-left: 50px“>
	<span *style*="text-align:center">
<font>	<span *style*= “font: 10px Arial red">
<dir>	
<menu>	
<s> <u>	<span *style*="text-decoration:underline">

3.2.2 Atribut

Sebuah *tag* memiliki Atribut yang menjelaskan ciri-ciri tentang elemen HTML dan Atribut tersebut selalu dituliskan di *tag* awal. Sebuah tag HTML dapat memiliki lebih dari satu Atribut. Jenis Atribut dapat dikelompokkan ke dalam 4 kategori, yaitu:

a. Atribut Wajib: Atribut yang harus dimasukkan baru dapat memperlihatkan keberadaan *tag*

   Contoh: `<img src = “gambar.jpg">`

b. Atribut Komplemen: Atribut yang bersifat tambahan, artinya *tag* sudah terlihat meski Atribut ini tidak digunakan hanya kurang maksimal

   Contoh: `<img width = "10%">`

c. Atribut Definisi: Atribut yang terkait pendefinisian elemen dalam halaman web seperti contoh pada tabel 3.3.

   Contoh: `<a title="Link Ke Google"> Google </a>`

Tabel 3. 3 Atribut Definisi

Atribute	Value	Deskripsi
class	*Class name*	Mendefinisikan class
id	*Id*	Mendefinisikan Id
style	*Style definition*	Mendeskripsikan *style*
title	*text*	Memberi Keterangan

d. Atribut *Event*: Atribut yang digunakan untuk memanggil *script* event seperti contoh pada tabel 3.4

Contoh:

```
<a href = "http://google.com" onmouseover = "script()"> Google </a>
```

Tabel 3. 4 Atribut Event

Atribute	Value	Deskripsi
Body dan Frameset		
Onload	*script*	Dokumen di-load
Onunload	*script*	Dokumen di-unload
Form		
onblur	*script*	Elemen kehilangan focus
onchange	*script*	Kondisi elemen berubah
onfocus	*script*	Kondisi elemen fokus
onreset	*script*	Form di-reset
onselect	*script*	Kondisi elemen terpilih
onsubmit	*Script*	Form di-submit
Event Img (image)		
onabort	*script*	Loading gambar terinterupsi
Event Keyboard		
onkeydown	*script*	Tuts ditekan
onkeypress	*script*	Tuts ditekan dan dilepas
onkeyup	*script*	Tuts dilepas
Event Mouse		
onclick	*script*	Mouse diklik

ondblclick	*script*	Mouse diklik 2 kali
onmousedown	*script*	Mouse diklik-tahan
onmousemove	*script*	Kursor mouse bergerak
onmouseout	*script*	Kursor mouse berpindah
onmouseover	*script*	Kursor mouse bergerak di atas elemen
onmouseup	*script*	Mouse dilepas

Seperti pada *tag*, beberapa Atribut di HTML4 mengalami perubahan di HTML5. Mayoritas Atribut HTML beralih menjadi *style* CSS, serta terkesan semakin rumit, misalnya `<span style="text-decoration:underline;">..</span>` yang menggantikan `<u></u>`. Namun, tujuan utama penyesuian ini adalah pengaturan tampilan halaman lebih komprehensif, yaitu bahwa garis bawah (*underline*) termasuk dalam pengaturan teks. Model penulisan elemen seperti ini akan sangat membantu ketika kita ingin memodifikasi elemen yang sudah ada, misalnya: kita dapat mengganti elemen h3 dengan versi kita sendiri, ebagai contoh:

```
h3{
    font-size: 10pt; font-weight: bold; text-
    decoration: underline;
}
```

Style pengganti lainnya dapat dilihat pada tabel 3.5.

Tabel 3. 5 *Style* CSS Pengganti Atribut Usang HTML4

Atribut Usang	Elemen	*Style* CSS Pengganti *)
Align	<img>, <table>, , <h1..6>, <p>, <hr>	`img {float:center;}`
Alink	<body>	`a:link{color:red;}`

Background	<body>	`body{background-image:foo.jpg;}`
Bgcolor	<table>,<tr>,<td>, <th>	`td{background-color:#000;}`
Compact	,	`ul{list-style: compact;}`
Color	<basefont>, <font>	`div{text-color:#000000}`
Border	<img>,<object>	`Img{border:1px solid;}`
Hspace	<img>,<object>	`Img{padding-left:5px}`
Vspace	<img>,<object>	`Img{padding-top:5px}`
Link	<body>	`<link>`
Name	<form>, <button>	`Id`
Noshade	<hr>	`hr{shadow:none;}`
Nowrap	<td>, <th>	`td,th{white-space:nowrap;}`
Size	<basefont>,	`p {font-size:10px}`
Text	<body>	`body {color: red}`
Type		`ol {list-style-type: circle}`
Value		`ol {list-style: georgian}`
Vlink	<body>	`a:visited{color:red}`
Width	<hr>,<pre>,<td>, <th>	`hr {width:100px}`

*Penulisan Atribut *style* menjadi properti CSS dibahas pada bab 4

3.2.3 Penulisan HTML

Penulisan elemen HTML pada gambar 3.2 tidak sesuai standar bahasa *markup* yang baik. Kesalahan tersebut antara lain:

- Tidak terdapat deklarasi dokumen html
- Terdapat *tag* pembuka <li> namun tidak ditutup.
- *Tag* menggunakan huruf kapital.

Adapun perbaikan penulisan dapat dilihat pada gambar 3.3. Ada beberapa kebiasaan lama yang perlu diubah ketika menulis dokumen HTML antara lain:

1. Elemen html, body, head dan title seharusnya ada
2. Semua *tag* dan Atribut ditulis dengan huruf kecil

   <A HREF="foo.html">...</A> kurang tepat

   ... tepat
3. *Tag* Kosong boleh dituliskan dengan menambahkan tanda slash (/), tetapi boleh juga tidak

 <img src="foo.jpg"> tepat

 dan tepat

```
<!DOCTYPE html>
<html lang="en-us">
  <head>
      <title>Document Title</title>
  </head>
  <body>
    <h1>Heading Utama</h1>
    <h2>SubHeading</h2>
    <p>Text dalam sebuah paragraf</p>

    <ul>
          <li> List Bullet Pertama </li>
          <li> List Bullet Kedua </li>
    </ul>
  </body>
```

Gambar 3. 3 Perbaikan HTML Gambar 3.2

4. *Tag* tutup tidak boleh dihilangkan.

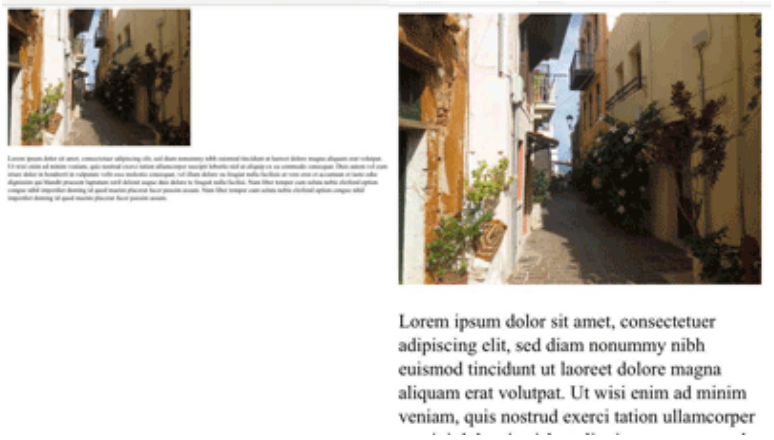
    teks paragraf kurang tepat

    teks paragraf tepat
5. *Value* dari sebuah Atribut harus diapit oleh tanda petik (“).

   <hr size=2> kurang tepat

<hr size="2" > tepat

6. Semua *tag* harus tersusun dengan urutan pembuka dan penutup yang sesuai.
    Teks ini dicetak tebal dan miring kurang tepat
    Teks ini dicetak tebal dan miring tepat
7. Penggunaan spasi dan =
   <link rel = "*style*sheet" href = "*style*s.css"> kurang tepat
   <link rel="*style*sheet" href="*style*s.css"> tepat
8. Meta data untuk tipe *encoding* didefinisikan. *Charset* UTF-8 melingkupi semua karakter dan simbol dari berbagai negara, jadi *website* berbahasa korea pun bisa dikenali browser.
   <meta charset="UTF-8">
9. Meta data skala ukuran tampilan dan perangkat didefinisikan <meta name="viewport" content="width=device-width, initial-scale=1.0">. Tampilan tanpa dan dengan *viewport* dapat dilihat pada gambar 3.4.



(a) Tanpa *viewport* (b) Dengan *viewport*

Gambar 3.4 Penggunaan *Viewport*

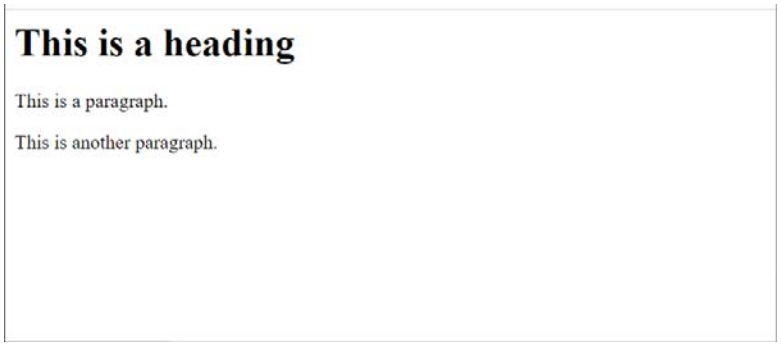
3.2.4 Elemen-elemen HTML Penting

Dalam penulisannya, dokumen HTML selalui diawali dengan *tag* <html> dan diakhiri dengan </html>. Bagian yang terlihat dari dokumen HTML adalah antara <body> dan </body>, sementara bagian yang tidak terlihat (bersifat meta data) dituliskan dalam *tag* <head> dan </head>

- **Title**

```
<html>
      <head>
          <title>Page title</title>
      </head>
      <body>
          <h1>This is a heading</h1>
          <p>This is a paragraph.</p>
          <p>This is another paragraph.</p>
      </body>
</html>
```

Tampilan:



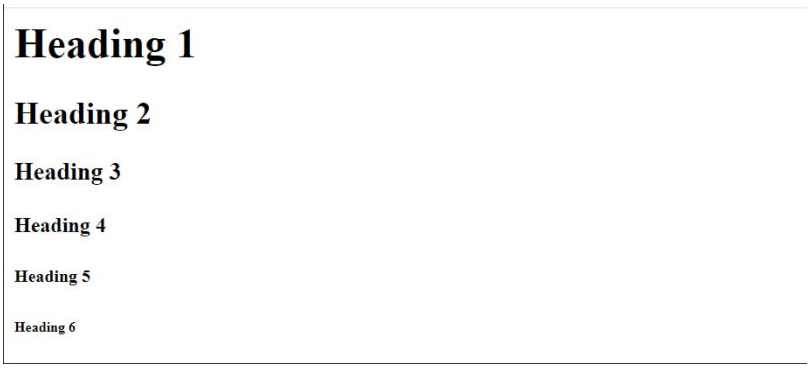
- ***Heading***

Elemen *Heading* adalah judul atau subjudul yang ingin Anda tampilkan di halaman web. *Heading* didefinisikan dengan *tag* <h1> hingga <h6>. <h1> mendefinisikan *heading* yang paling penting atau paling

besar dan <h6> mendefinisikan *heading* yang paling tidak penting paling kecil. Perhatikan kode *Heading* dibawah ini:

```
<html>
      <body>

         <h1>Heading 1</h1>
         <h2>Heading 2</h2>
         <h3>Heading 3</h3>
         <h4>Heading 4</h4>
         <h5>Heading 5</h5>
         <h6>Heading 6</h6>

      </body>
</html>
```

Tampilan:

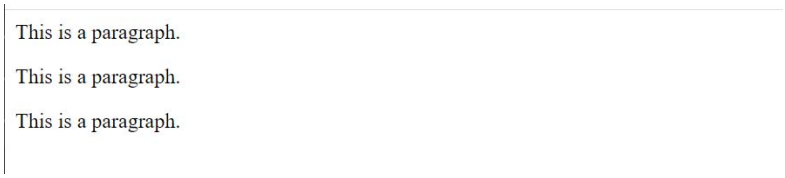


- **Paragraf**

Paragraf selalu dimulai pada baris baru, dan biasanya berupa blok teks. Elemen **<p>** dapat digunakan untuk membuat sebuah paragraf, browser secara otomatis akan menambahkan beberapa spasi (margin) sebelum dan sesudah paragraf. Contoh:

```
<html>
      <body>

          <p>This is a paragraph.</p>
          <p>This is a paragraph.</p>
          <p>This is a paragraph.</p>

      </body>
</html>
```

Tampilan:



- **List Item**

List Item pada HTML memungkinkan pengembang web untuk mengelompokkan satu set item terkait dalam daftar. List item dapat dibagi menjadi dua yaitu:

- ***Unordered List***

  Unordered List dimulai dengan *tag* **<ul>**. Setiap item daftar dimulai dengan tag **<li>**. List Item akan ditandai dengan lingkaran hitam kecil. Contoh:

```
<html>
      <body>

      <h2>An unordered HTML list</h2>

      <ul>
        <li>Coffee</li>
        <li>Tea</li>
        <li>Milk</li>
      </ul>

      </body>
</html>
```

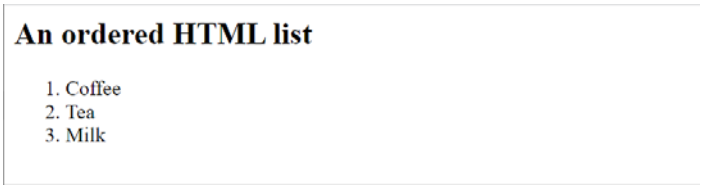
Tampilan:

- ***Ordered List***

Ordered List dimulai dengan *tag* **<ol>**. Setiap item daftar dimulai dengan *tag* **<li>**. List Item akan ditandai dengan angka.

```
<html>
      <body>

          <h2>An ordered HTML list</h2>

          <ol>
            <li>Coffee</li>
            <li>Tea</li>
            <li>Milk</li>
          </ol>

      </body>
</html>
```

Tampilan:

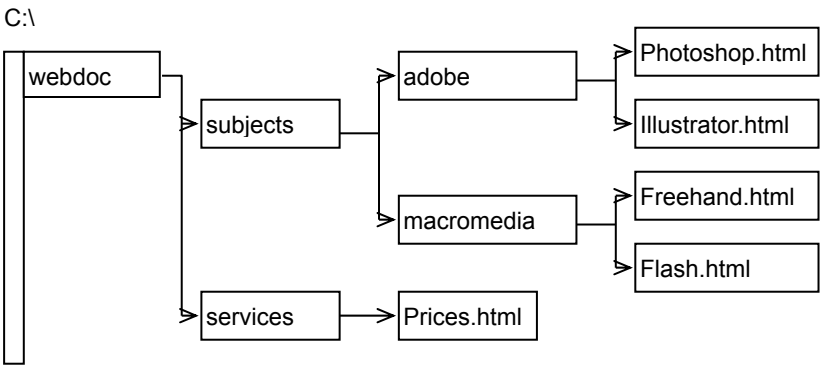


- *Link Anchor*

Anchor merupakan elemen penting pada web yang ditemukan di hampir semua halaman web. *Links* dapat merujuk ke dokumen lain dalam *website* yang sama ataupun ke *website* yang berbeda. *Link* memungkinkan pengguna

bernavigasi dari halaman ke halaman. Jika *mouse* berada di atas tautan, kursor akan berubah menjadi tangan. Untuk membuat *Link*, gunakan *tag* **<a>** dan atribut **href** yang menunjukkan target tautan. Klik *link* untuk membuka target halaman yang dituju.

Ada 2 cara menuliskan path *file* tautan, yaitu secara absolut dan relatif. Path absolut berdasar pada lokasi yang statis, sementara *path* relatif berpatokan pada lokasi dokumen sekarang.



Pada struktur direktori webdoc di atas, jika ingin membuat link ke halaman illustrator dari dokumen di webdoc, gunakan path relatif seperti berikut::

```
<a href="subjects/adobe/illustrator.html">
```

Tidak perlu gunakan path absolute seperti berikut:

```
"C:\webdocs\subject\adobe\illustrator.html"
```

Jika ingin membuat link ke folder satu level di atasnya, misalnya tautan ke halaman Flash dari halaman Photoshop, ketikkan:

```
<a href=../macromedia/flash.html>
```

Dua titik menandakan satu folder di atasnya, diikuti dengan folder dan nama *file* flash.html (halaman tujuan).

Jika ingin membuat link ke halaman Prices pada folder *Services* dari halaman Photoshop, ketikkan:

```
<a href=../../services/prices.html>
```

Dua tanda titik menandakan *file* target berada di folder dua level di atasnya

Dalam sebuah dokumen yang panjang, kadang dibutuhkan link ke beberapa bagian halaman misalnya:

```
<a href="#atas">
```

Target berada di dalam dokumen yang sama dengan nama "atas". Untuk berpindah ke bagian tertentu saja dari target halaman lain, gunakan:

```
<a href="foo.html#bawah">
```

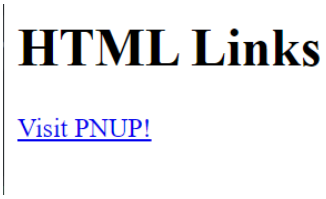
Artinya *link* ke lokasi dokumen foo.html bagian bawah

Agar bagian target dikenali, maka target harus diberi nama misalnya

Contoh penulisan *link* sederhana:

```
<html>
 <body>
   <h1>HTML Links</h1>
   <p>
     <a href="https://poliupg.ac.id">
        Visit PNUP!
     </a>
   </p>
 </body>
</html>
```

Tampilan:



- ***Image***

Gambar pada web dapat meningkatkan nilai tampilan sebuah halaman web. Untuk memasukkan sebuah gambar, gunakan *tag* HTML **<img>**. Berbeda dari *tag* HTML lainnya, **<img>** adalah *tag* kosong. Untuk memasukkan gambar pada **<img>** maka lokasi gambar perlu di masukkan dalam atribut **src**.

```
<html>
  <body>
    <h2>Image</h2>
       <img src="img_chania.jpg" width="460"
       height="345">
  </body>
</html>
```

*)Width dan height merupakan properti CSS, tetapi di HTML5, elemen <canvas>, <embed>, <iframe>, <img>, <input>, <object>, dan <video> juga memiliki Atribut width dan height

Tampilan:

Image

- **Table**

Tabel dapat mengatur data ke dalam baris dan kolom. Untuk membuat tabel, Anda dapat menggunakan *tag* **<table>**. Kemudian, isi dari sebuah tabel dalam HTML terdiri dari sel-sel yang tersusun dalam baris dan kolom. Kolom selalu menjadi bagian sebuah baris. Untuk mendefinisikan baris dapat dilakukan dengan tag **<tr>** dan kolom didefinisikan dengan **<td>**. Contoh:

```
<html>
   <body>

      <h2>A basic HTML table</h2>
      <table border='1'>
        <tr>
          <th>Company</th>
          <th>Contact</th>
          <th>Country</th>
        </tr>
        <tr>
```

```
          <td>Alfreds Futterkiste</td>
          <td>Maria Anders</td>
          <td>Germany</td>
        </tr>
        <tr>
          <td>Centro comercial Moctezuma</td>
          <td>Francisco Chang</td>
          <td>Mexico</td>
        </tr>
      </table>

   </body>
</html>
```

Tampilan:

A basic HTML table

Company	Contact	Country
Alfreds Futterkiste	Maria Anders	Germany
Centro comercial Moctezuma	Francisco Chang	Mexico

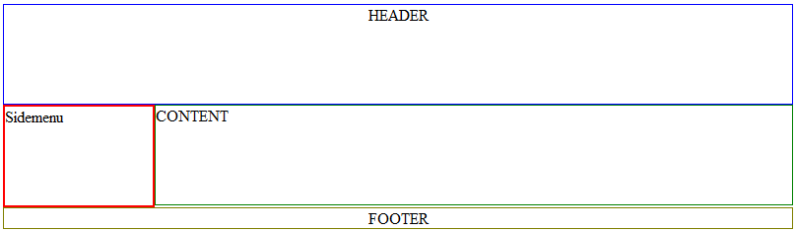
3.2.5 Layout Halaman

Elemen div sangat penting perannya dalam layout halaman. Meski di HTML5 telah ada elemen semantik yang mengatur layout, tetapi div masih sangat dibutuhkan untuk mengelola pembagian *section* sebuah halaman. Contohnya:

```
<html>
   <body>
      <div style ="border:1px solid blue;
      width:800px; height:100px; text-
      align:center">  HEADER </div>

      <div style="float:left; width:150px;
      border:2px solid red;
      height:100px;">Sidemenu </div>

      <div style="float:left; width:646px;
      border:1px solid green; height:100px;">
      CONTENT </div>
```

```
      <div style ="clear:both; border:1px solid
      olive; width:800px; text-align:center;">
      FOOTER </div>
   </body>
</html>
```

Tampilan:



Selain menggunakan elemen div, kita juga dapat menggunakan elemen semantik pada HTML5, yaitu:

<header> - Bagian *header* dokumen atau *section*

<nav> - Bagian *link* navigasi/ menu

<section> - Bagian pada sebuah dokumen

<article> - Bagian isi

<aside> - Bagian sisi isi berfungsi sebagai *sidebar*

<footer> - Bagian *footer* dokumen atau *section*

Contoh penggunaan elemen layout semantik:

```
<html>
   <body>
      <header>
        <h2>Cities</h2>
      </header>

      <section>
        <nav>
          <ul>
```

```
            <li><a href="#">London</a></li>
            <li><a href="#">Paris</a></li>
            <li><a href="#">Tokyo</a></li>
          </ul>
        </nav>

        <article>
          <h1>London</h1>
          <p>London is the capital city of
          England. It is the most populous city
          in the United Kingdom, with a
          metropolitan area of over 13 million
          inhabitants.</p>
        </article>
      </section>

      <footer>
        <p>Footer</p>
      </footer>

   </body>
</html>
```

Elemen semantik tidak dibuat untuk menggantikan elemen div, melainkan menguatkan penamaan sesuai dengan maknanya, tetapi tidak secara otomatis memiliki *style*, sehingga elemen-elemen ini tetap perlu diberi *style* seperti pada div, seperti berikut ini:

```
<style>
   * {
     box-sizing: border-box;
   }

   /* Style header */
   header {
     background-color: #666;
     padding: 30px;
     text-align: center;
     font-size: 35px;
     color: white;
   }

   /* Kolom navigasi bagian kiri */
```

```
    nav {
      float: left;
      width: 30%;
      background: #ccc;
      padding: 20px;
    }

    /* Style list dalam menu */
    nav ul {
      list-style-type: none;
      padding: 0;
    }

    article {
      float: left;
      padding: 20px;
      width: 70%;
      background-color: #f1f1f1;
    }

    /* Clear floats after the columns */
    section::after {
      content: "";
      display: table;
      clear: both;
    }

    /* Style the footer */
    footer {
      background-color: #777;
      padding: 10px;
      text-align: center;
      color: white;
    }

    /* Style responsif untuk nav dan article */
    @media (max-width: 600px) {
      nav, article {
        width: 100%;
        height: auto;
      }
    }
</style>
```

*) Penulisan *style* model ini akan dijelaskan pada bab 4

Tampilan:

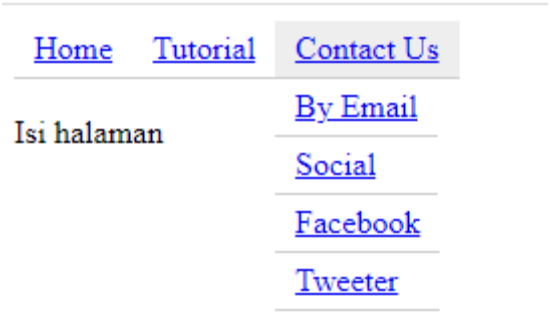
3.2.6 Navigasi

Navigasi dalam halaman web dibangun menggunakan HTML dan CSS. Sebagai contoh:

```
<style>
   .clear{clear:both;}
   ul{list-style-type:none; margin:0;padding:0;}
   ul#main li{background:white;float:left;}
   ul#main li a{display:block;padding:5px 10px;
   border-bottom:1px solid #ccc;}
   ul#main li a:hover{background:#eee;}
   ul#main li ul li{float:none;}
   ul#main li ul {position:absolute;display:none;}
   ul#main li ul li
   ul{position:absolute;display:none;}
   ul#main li:hover ul{display:block;}
</style>
<div>
  <ul id="main" style="background:red;">
     <li><a href="#">Home</a></li>
     <li><a href="#">Tutorial</a>
         <ul>
          <li><a href="#">Java</a></li>
          <li><a href="#">PHP</a></li>
          <li><a href="#">Delphi</a></li>
         </ul>
     </li>
     <li><a href="#">Contact Us</a>
       <ul>
```

```
          <li><a href="#">By Email</a></li>
          <li><a href="#">Social </a>
             <ul>
               <li><a href="#">Facebook</a></li>
               <li><a href="#">Tweeter</a></li>
             </ul>
          </li>
        </ul>
      </li>
  </ul>
</div>

<div class="clear">
  <p><br/>Isi halaman </p>
</div>
```

Tampilan:



*) *Style* menggunakan CSS akan dijelaskan secara rinci pada bab berikutnya.

3.3. Rangkuman

Hypertext Markup Language atau HTML adalah bahasa markup yang digunakan untuk membuat sebuah halaman web. Isinya terdiri dari berbagai elemen yang membentuk struktur sebuah halaman web. Beberapa tag yang sering digunakan ialah:

1. <html> </html> sebagai pembuka halaman HTML
2. <h1> - <h6> sebagai *heading* halaman

3. sebagai paragraf teks halaman
4. / sebagai list item
5. <a> sebagai *link*
6. <img> sebagai pengisi gambar
7. <table> sebagai pembuat *table*
8. sebagai *layout* halaman
9. , <a> sebagai navigasi

3.4. Tugas Latihan

1. Buatlah *protype* untuk *homepage* halaman web sekolah yang berisi profil sekolah menggunakan *tag-tag* HTML paragraf, list item, *link*, gambar dan *table*.
2. Buatlah sebuah dokumen HTML sebagai berikut

```
<body alink='style.css'>
   <p><font  size="10">Paragraf  dengan  ukuran
   font 10</font></p>
   <ul type="circle">
      <li>List pertama</li>
      <li>List kedua </li>
    </ul>
    <table cellspacing="10">
      <tr>
        <td width="100">Kolom pertama</td>
        <td width="200">Kolom kedua</td>
      </tr>
    </table>
</body>
```

3. Ubahlah HTML nomor 2 agar sesuai dengan standar HTML5
4. Buatlah dokumen HTML untuk tabel berikut:

		Siswa	
		Laki-laki	Perempuan
Kelas	1	501	480
	2	500	470

5. Buatlah navigasi menu berisi 3 menu yaitu Beranda, Produk, dan Tentang Kami. Pada menu produk terdapat kategori Buku, Novel, dan Komik.

3.5. Pustaka

1. Ducket, J., HTML & CSS Design and Build *Website*s, John Wiley & Sons, 2011
2. Hogan, B.P., HTML5 & CSS3 Develop with Tomorrows Standards Today, USA, Pragmatic *Programmer*s, 2010
3. Sari, A.O., dkk, Web Programming, Graha Ilmu, Jakarta, 2019
4. https://www.w3schools.com/html

BAB 4
HTML Form

HTML *Form* digunakan untuk mengambil input dari pengguna. HTML *form* sangat penting dalam pengembangan web karena merupakan jalur utama data dari *user* ke aplikasi.

Setelah menyelesaikan bab ini, Anda diharapkan dapat:

1. Memahami cara membuat *form*
2. Menuliskan kode untuk pengiriman data pada *form* HTML
3. Memahami cara menangkap data dari *form* HTML

4.1. Pengantar Materi

Formulir paling banyak digunakan untuk membuat buku tamu atau umpan balik. Sebuah *form* dalam HTML di definisikan menggunakan *tag* HTML <form> dan di tutup dengan </form>

```
<form>
      .
      form elements
      .
</form>
```

Dalam sebuah *form* kemudian beberapa elemen pendukung untuk input di berikan seperti <input>, <label>, <textarea>, dan lainnya.

4.2. Isi Materi

4.2.1. Elemen *Form*

Form memiliki beberapa Atribut wajib yang dapat dituliskan sebagai berikut:

<FORM METHOD=”post” ACTION=”proses.php”>…</FORM>

Memberitahu cara informasi dikirim

Memberitahu server tujuan informasi akan dikirim

Form HTML harus di dukung oleh beberapa atribut wajib, yaitu:

1. Atribut action=’’

   Atribut action mendifinisikan tujuan pengiriman data dalam sebuah form. Umumnya, action mengarahkan pengiriman data pada sebuah link / *file* khusus pada server yang berfungsi sebagai penangkap data dan pengolah data.

```
<form action=”aksi_insert.php”>
      .
      form elements
      .
</form>
```

2. Atribut method=’’

Atribut method digunakan untuk mendefinisikan cara sebuah form dalam mengirim data. Terdapat dua cara yang bias digunakan yaiitu method POST dan method GET. Metode GET merupakan metode *default* pengiriman data sebuah *form*. Data dikirimkan ke server dengan meletakkan data pada bagian akhir

URL. Sebagai contoh, URL *action* menunjuk ke proses.php dan terdapat elemen dengan nama "kota" dengan nilai "Jakarta" dan "telepon" dengan nilai "2503645", maka URL akhir yang dikirim ke server adalah `proses.php?kota=Jakarta&telepon=2503645.`

Tidak seperti metode GET, metode POST mengirimkan data secara terpisah dari URL dalam format standar input. *Script* mengambil data form dari input dan dapat menyimpan data dalam jumlah banyak menggunakan struktur data *array*. Data yang dikirim lebih aman karena tidak tampak pada URL. Metode POST cocok digunakan untuk mengiriman data besar dan *file* sedangkan GET digunakan untuk mengiriman data form yang ringkas melalui URL.

```
<form action=″aksi_insert.php″ method=″POST″>
      .
      form elements
      .
</form>
```

3. Atribut enctype=''

   Enctype merupkan atribut optional pada *form* yang tidak selalu di butuhkan. *Enctype* baru akan dibutuhkan jika, form di rancang dengan pilihan upload *file*.

```
<form action=″aksi_insert.php″ method=″POST″
enctype=″multipart/form-data″>
      .
      form elements
      .
</form>
```

Terdapat juga Atribut tambahan seperti:

- Id, nama identitas yang dapat digunakan pada saat menggunakan Javascript() dengan fungsi getElementById().
- Name, nama *form* yang digunakan pada saat melakukan pemrosesan data pada form.

4.2.2. Bagian *Form*

Pada *form*, terdapat elemen-elemen khusus yang dapat digunakan untuk membuat berbagai macam bentuk input dari *user*, seperti pada tabel 3.1. Beberapa elemen yang umum digunakan yaitu:

- <input>

  Salah satu elemen *form* yang paling sering digunakan adalah elemen <input>. Elemen <input> dapat ditampilkan dalam beberapa cara, tergantung pada atribut *type*.

```
<html>
  <body>

     <h2>The input Element</h2>

     <form action="/action_page.php">
         <label for="fname">First name:</label><br>
         <input type="text" id="fname"
         name="fname"><br><br>
         <input type="submit" value="Submit">
     </form>

  </body>
</html>
```

Tampilan: :

Tabel 4. 1 Elemen-elemen *Form*

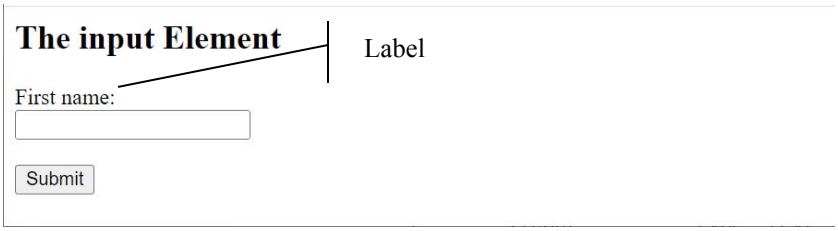
Elemen Form	Keterangan	Tampilan Halaman
TEXTBOX What's your name? `<INPUT TYPE="text" NAME="email" />`	Tipe input textbox dengan nama "email"	What's your email ? 
TEXT AREA Have you got anything to tell me? `<TEXTAREA NAME="comment" ROWS="13" COLS="50"> </TEXTAREA>`	Nama textarea "comment" dengan jumlah baris 13 dan jumlah kolom 50 karakter	Have you got anything to tell me? 
DROP DOWN BOX What's your favorite color? `<select name="favcolor"> <option selected>red <option>orange <option>yellow`	Memulai drop down box dengan nama "favcolor", dengan opsi default "red"	What's your favorite color? 

<option>green <option>gray</option> </select> 		
CHECK BOXES I enjoy (check all that apply): <INPUT TYPE="checkbox" NAME="fav"> sports <INPUT TYPE="checkbox" NAME="movies"> horror movies	Tipe input adalah checkbox dengan nama "fav"	I enjoy (check all that apply): ☑ sports ☐ horror movies
RADIO BUTTONS What is your Gender? <INPUT TYPE="radio" NAME="gender" VALUE="male" CHECKED> MALE <INPUT TYPE="radio" NAME="gender" VALUE="female"> FEMALE	Input bertipe radio dengan nama "gender" dengan nilai "male" dan "female" dengan nilai default yang ter-checked adalah male	What is your Gender? ◉ MALE ○ FEMALE
SUBMIT BUTTON <INPUT TYPE="submit" VALUE="Mail My	Tipe input submit dengan nilai "Mail My Comments"	Mail My comments Clear Form

comments> <INPUT TYPE=”reset” VALUE=”Clear Form>	dengan event onclick submit isi form Tipe input reset dengan nilai “Clear” dengan event onclick menghapus isi form	

- <label>

  Elemen <label> mendefinisikan label untuk beberapa elemen form. Elemen <label> berguna bagi *user* untuk memperjelas apa yang harus diisikan pada bagian input tertentu.



- <select> dan <option>

  Elemen <select> digunakan untuk membuat input pilihan.

```
<html>
   <body>
      <p>The select element defines a drop-down
      list:</p>

      <form action="/action_page.php">
          <label for="cars">Choose a car:</label>
          <select id="cars" name="cars">
              <option value="volvo">Volvo</option>
```

```
            <option value="saab">Saab</option>
        </select>
        <input type="submit">
    </form>
</body>
</html>
```

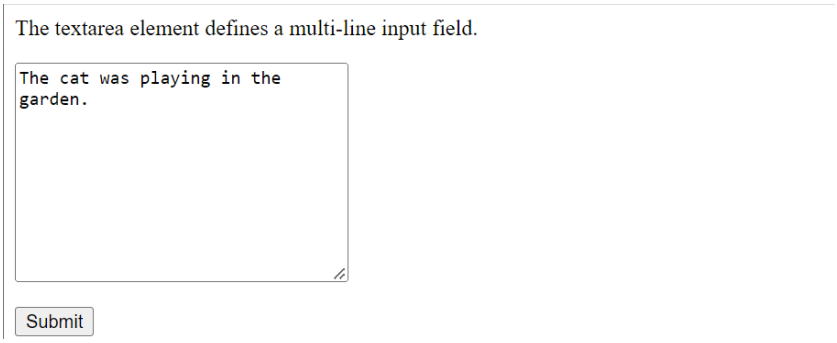
Tampilan:

- <textarea>

  Element <textarea> merupakan elemen yang umum digunakan untuk input yang memiliki sifat panjang.

```
<html>
  <body>

    <p>The textarea element defines a multi-line
    input field.</p>

    <form action="/action_page.php">
        <textarea name="message" rows="10"
        cols="30"> The cat was playing in the
        garden.</textarea>
        <br><br>
        <input type="submit">
    </form>

  </body>
</html>
```

Tampilan:



4.2.3. Elemen Input

Elemen <input> pada *form* web merupakan elemen yang special karena memiliki beberapa bentuk *type*. Beberapa *type* input yang umum digunakan yaitu:

1. Tipe Text

   Input *type text* merupakan input yang diperuntukkan untuk menerima masukan dari *user* berupa text.

```
<html>
  <body>

  <h2>Text field</h2>

  <form action="/action_page.php">
    <label for="fname">First name:</label><br>
    <input type="text" id="fname" name="fname"><br>
    <label for="lname">Last name:</label><br>
    <input          type="text"          id="lname"
  name="lname"><br><br>

    <input type="submit" value="Submit">
  </form>

  </body>
</html>
```

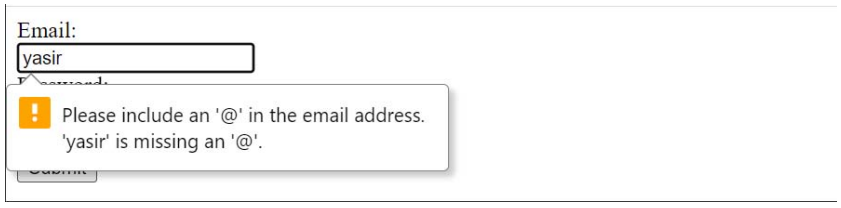
Tampilan:

2. Tipe Email dan *Password*

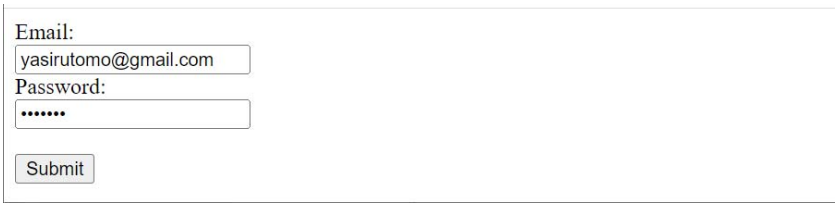
   Type password umumnya digunakan untuk menerima masukan dari *user* terkait password, sedangkan email dikhususkan hanya menerima input berbentuk format email.

```
<html>
   <body>

      <form action="/action_page.php">
          <label for="email">Email:</label><br>
          <input type="email" id="emal"
          name="email"><br>

          <label for="pwd">Password:</label><br>
          <input type="password" id="pwd"
          name="pwd"><br><br>

          <input type="submit" value="Submit">
      </form>

   </body>
</html>
```

Jika bukan format email yang diinput, tampilannya:



Tampilan input password:



3. Tipe *file*

   Tipe *file* digunakan untuk menerima input *user* dalam bentuk *file*.

```
<html>
   <body>

      <form method="post" enctype="multipart/form-
      data">
          <label for="file">File:</label><br>
          <input type="file" name="file"><br><br>
          <input type="submit" value="Submit">
      </form>

</body>
</html>
```

Tampilan:

4. Tipe Submit

   Type submit digunakan sebagai tombol untuk mengirim data.

```
<input type="submit" value="Submit">
```

Tampilan:

4.2.4. Metode Pengiriman Data

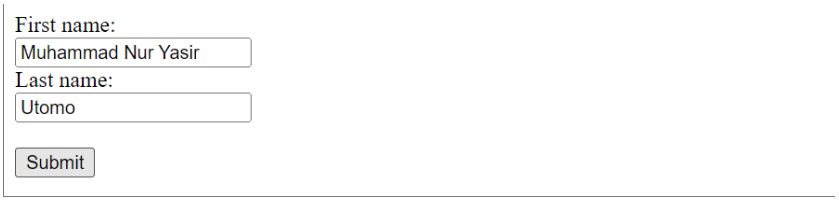
Pada bagian Atribut *form* telah dijelaskan tentang metode GET dan POST. Berikut contoh penggunaannya:

1. Pengiriman dengan metode POST

   Form mengirim data melalui protokol HTTP (data yang dikirim tidak ditampilkan di URL).

```
<html>
   <body>

      <form action="tangkap.php" method="post">
        <label for="fname">First name:</label><br>
        <input type="text" name="fname"><br>

        <label for="lname">Last name:</label><br>
        <input type="text" name="lname"><br><br>

        <input type="submit" value="Submit">
      </form>

   </body>
</html>
```

Tampilan:



Untuk menangkap data *form* yang dikirm dapat menggunakan *super global variable* $_POST dengan cara berikut:

```
<?php
        echo $_POST['fname'];
        echo '<br>';
        echo $_POST['lname'];
?>
```

Tampilan:

2. Method GET

Method GET mengirim data melalui URL dengan cara memasangkan nama input dan nilainya. Nilai dalam GET dibatasi 2048 karakter. Karena data yang dikirim tampil pada URL maka, tidak disarankan untuk digunakan dalam mengirim data sensitif.

```
URL:
http://localhost/coba/tangkap.php?nama=Yasir%20Utomo&nim=24
```

Untuk menangkap data dapat menggunakan *super global*

variable $_GET dengan cara berikut:

```
<?php
   echo $_GET['nama'];
   echo '<br>';
   echo $_GET['nim'];
?>
```

Tampilan:

Yasir Utomo
24

Untuk melakukan pengiriman data form, diperlukan bahasa pemrograman di sisi server yang akan menerima data data *client* melalui inputan *form*. Bahasa pemrograman ini akan dibahas pada bab 8 pada bagian fungsi PHP *Form*. Sebelum dikirim ke server, data form dapat pula dimanipulasi di sisi *client*, dengan melakukan validasi terlebih dahulu sebelum data dikirim ke server. Bahasa pemrograman di sisi *client* dapat menggunakan Javascript. Bahasa Javascript dibahas pada bab 6.

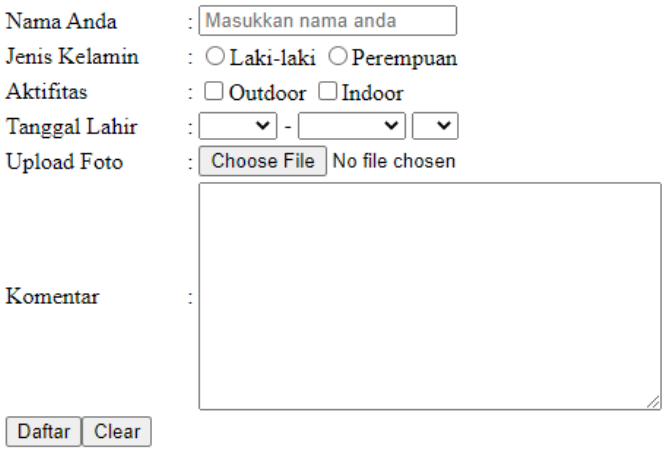
4.3. Rangkuman

Web *Form* merupakan cara untuk membuat formulir input untuk kebutuhan menerima masukan *user*. Web *Form* memiliki beberapa atribut khusus seperti *action, method* dan *enctype*. Terdapat beberapa *tag* elemen form yang mendukung sebuah form seperti <input>, <textarea>, <select >, dan lainnya. Form dapat dikirim dengan menggunakan dua cara yaitu method POST dan GET. Agar dapat diterima di server, data yang dikirim

dari *client* perlu dikelola memanfaatkan bahasa pemrograman web.

4.4. Tugas Latihan

1. Sebutkan minimal 5 elemen inputan dalam *form*
2. Jelaskan perbedaan jika form dikirim dengan metode get dan post
3. Apa saja bahasa pemrograman yang diperlukan untuk mengelola data yang berasal dari elemen-elemen *form*?
4. Buatlah halaman *form* berikut:



5. Tangkap semua isian dalam form menggunakan method GET. Tampilkan URL yang dihasilkan beserta tampilan data dari formulir
6. Buat sebuah Web Form yang tujukan sebagai halaman Kontak *website* menggunakan minimal 5 jenis elemen form yang berbeda.

4.5. Pustaka

1. Hogan, B.P., HTML5 & CSS3 Develop with Tomorrows Standards Today, USA, Pragmatic *Programmer*s, 2010
2. Ducket, J., HTML & CSS Design and Build *Website*s, John Wiley & Sons, 2011
3. Sari, A.O., dkk, Web Programming, Graha Ilmu, Jakarta, 2019
4. https://www.w3schools.com/html/html_forms.asp

BAB 5
CSS (Cascading *Style* Sheets)

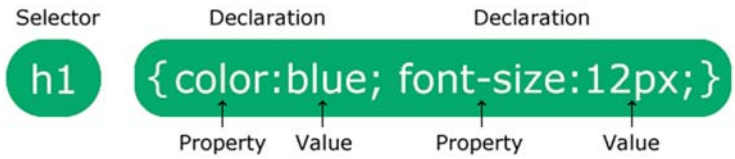
CSS atau Cascading *Style* Sheet adalah bahasa yang kita gunakan untuk menata halaman web termasuk *style* dan *layout* komponen web. CSS digunakan untuk menjelaskan bagaimana struktur elemen HTML ditampilkan di Browser.

Setelah menyelesaikan *bab* ini, Anda diharapkan dapat:

1. Mengetahui cara-cara menulis CSS
2. Menggunakan *style* CSS untuk desain web
3. Membuat tampilan antar muka (*user interface*) web yang baik

5.1. Pengantar Materi

Pada dasarnya CSS terdiri dari Selector dan blok deklarasi seperti di bawah ini:



Penjelasan sebagai berikut:

1. *Selector* digunakan sebagai penunjuk ke elemen HTML yang ingin diberikan *style* atau desain.
2. Blok deklarasi berisi satu atau lebih deklarasi *style* yang dipisahkan oleh titik koma.

3. Setiap deklarasi menyertakan nama properti CSS dan nilainya, dipisahkan oleh titik dua.
4. Beberapa deklarasi CSS dipisahkan dengan titik koma, dan blok deklarasi dikelilingi oleh kurung kurawal.

5.2. Isi Materi

Dalam mengatur *style* sebuah halaman, ukuran sangat penting untuk menunjukkan proporsianalitas halaman. Standar satuan yang digunakan ada yang bersifat absolut dan relatif. Satuan yang bersifat absolut memiliki standar baku, seperti berikut:

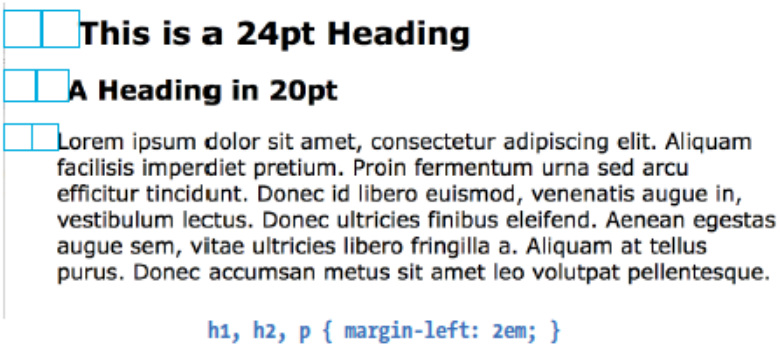
Satuan Absolut	Keterangan
px (pixel)	1/96 inchi
in (inchi)	
mm (millimeter)	
cm (centimeter)	
q (quarter)	1/4 millimeter
pt (point)	1/72 inchi
pc (pica)	1 pica = 12 point atau 1/6 inchi

Sementara yang bersifat relatif, ukurannya bergantung ukuran komponen lainnya, seperti berikut:

Satuan Relatif	Keterangan
em	Sama dengan ukuran font yang digunakan
ex (x-height)	Sekitar tinggi huruf kecil “x” pada font yang digunakan
Rem (root em)	Sama dengan ukuran em html root

Ch (zero width)	Sama dengan lebar angka 0 pada font yang digunakan
Vw (viewport width unit)	Sama dengan 1/100 lebar viewport (browser window)
Vh (viewport height unit)	Sama dengan 1/100 tinggi
Vmin (viewport minimum unit)	Sama dengan nilai vw atau vh, mana yang lebih kecil
Vmax viewport maximum unit	Sama dengan nilai vw atau vh, mana yang lebih besar

Contoh:



Untuk melihat tampilan *style* halaman pada console masing-masing browser, dapat mengakses link berikut:

- Chrome DevTools (View → Developer → Developer Tools): developer.chrome.com/devtools
- Firefox (Tools → Web Developer): developer.mozilla.org/en-US/docs/Tools
- Microsoft Edge (open with F12 key): developer.microsoft.com/en-us/microsoft-edge/platform/documentation/

- Safari (Develop → Show Web Inspector) : developer.apple.com/library/content/documentation/AppleApplications/Conceptual/Safari_Developer_Guide/Introduction/Introduction.html
- Opera (View → Developer Tools → Opera Dragonfly): www.opera.com/dragonfly/

5.2.1. *Selector* CSS

Selector pada CSS merupakan komponen yang dapat digunakan untuk memilih elemen HTML yang ingin di ubah/*style*. *Selector* CSS dapat dibagi menjadi beberapa bagian yaitu:

1. *Selector* Elemen

   Selector elemen memilih bagian dari halaman web berdasarkan tag HTML. Contoh:

```
<html>
      <head>
      <style>
      p {
        text-align: center;
        color: red;
      }
      </style>
      </head>
      <body>

      <p>Hello World.</p>

      </body>
</html>
```

Tampilan:

Dari gambar diatas terlihat bahwa *Selector* yaitu **p** akan mempengaruhi *tag* elemen HTML **<p>**.

2. *Selector* Id

Selector Id menggunakan atribut id dari elemen HTML untuk memilih elemen tertentu. Id dari sebuah elemen sebaiknya unik di dalam sebuah halaman, jadi pemilih id digunakan untuk memilih satu elemen unik. Untuk memilih elemen dengan id tertentu, tulis karakter *hash* **(#)**, diikuti dengan id elemen. Contoh:

```
<html>
      <head>
      <style>
      #para1 {
        text-align: center;
        color: red;
      }
      </style>
      </head>
      <body>

      <p id="para1">Hello World!</p>
      <p>This paragraph is not affected by the
      style.</p>

      </body>
</html>
```

Tampilan:

Dari gambar di atas terlihat bahwa hanya salah satu elemen saja dengan atribut id yang sesuai saja yang berubah.

3. *Selector Class*

Selector Class memilih elemen HTML dengan atribut *class* tertentu. Untuk memilih elemen dengan *class* tertentu, tulis karakter titik (.), diikuti dengan nama kelas. Contoh:

```
<html>
  <head>
    <style>
      .center {
            text-align: center;
            color: red;
      }
    </style>
  </head>
<body>

  <h1  class="center">  Red  and  center-aligned
  heading</h1>
  <p   class="center">Red   and   center-aligned
  paragraph.</p>

</body>
</html>
```

Tampilan:

Dari gambar diatas terlihat bahwa semua elemen yang memiliki *class center* terdampak.

5.2.2. Penggunaan CSS

Terdapat beberapa metode dalam menggunakan CSS. Penggunaan CSS dapat dibagi menjadi tiga yaitu *Inline* CSS, *Internal* CSS dan *External* CSS. Masing-masing metode dapat dijelaskan sebagai berikut:

1. *Inline* CSS

   Inline CSS merupakan metode untuk memberikan *style* pada elemen tertentu pada halaman HTML dengan memberikan atribut ***style*=""** dalam *Tag* HTML. Contoh:

```
<html>
   <body>

      <h1 style="color:blue;text-
      align:center;">This is a heading</h1>
      <p style="color:red;">This is a
      paragraph.</p>

   </body>
</html>
```

Tampilan:

2. Internal CSS

   Internal CSS merupakan metode penggunaan CSS dengan mengolompokkan *style* CSS sebuah tag khusus yaitu ***<style></style>***. Umumnya CSS internal di tempatkan didalam **<head></head>**. Contoh:

```
<html>
      <head>
      <style>
      h1 {
        color: maroon;
        margin-left: 40px;
      }
      </style>
      </head>
      <body>

          <h1>This is a heading</h1>

      </body>
</html>
```

Tampilan:

3. External CSS

   External CSS merupakan metode penggunaan CSS dengan menempatkan *style* CSS pada *file* terpisah dari halaman

HTML. *File* CSS tersebut kemudian di panggil dengan menggunaan tag <link>. Contoh:

File style.css

```
body {
  background-color: lightblue;
}
h1 {
  color: navy;
  margin-left: 20px;
}
```

File index.html

```
<html>
<head>
<link rel="stylesheet" href="style.css">
</head>
<body>

    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>

</body>
</html>
```

Tampilan:

5.2.3. Properti CSS

- Color

  Fungsi dari *style* color adalah merubah warna tulisan pada halaman HTML. Warna yang digunakan dapat berupa kode warna RGB, HEX, HSL, RGBA atau HSLA. Contoh:

```
<html>
      <body>

      <h1 style="color:Red;">Tomato</h1>
      <h1 style="color:#ffa500;">Orange</h1>

      </body>
</html>
```

Tampilan:

- *Background*

Background digunakan untuk memberi efek background pada elemen dari halaman web. Terdapat beberapa properti untuk *background* yaitu *background-color* serta *opacity, background-image, background-repeat, background-attachment, background-position,* dan *background* (ringkasan). Contoh:

```
<html>
  <body>
     <h1 style= ”background:rgba(128, 0 , 0,
     0.3)> Tomato </h1>
     <h1 style="background-color:#ffa500;">
     Orange</h1>
  </body>
</html>
```

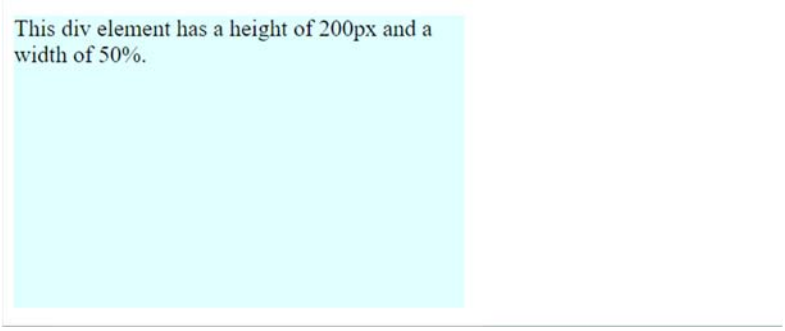
Tampilan:

- *Height and Width*

Properti *height* dan *width* digunakan untuk mengatur tinggi dan lebar suatu elemen. Nilai dari *height* dan *width* umumnya dapat berupa satuan px atau %. Contoh:

```
<html>
   <head>
   <style>
          div {
            height: 200px;
            width: 50%;
            background-color: powderblue;
          }
   </style>
</head>
<body>

  <div>This div element has a height of 200px
  and a width of 50%.</div>

</body>
</html>
```

Tampilan:



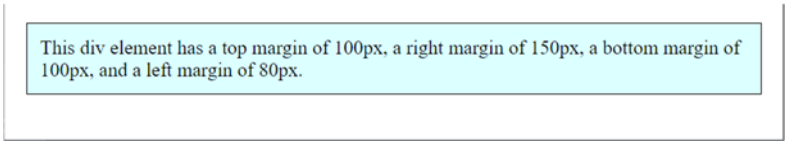
- *Margin and Padding*

Properti *margin* digunakan untuk memberikan jarak antar elemen, sedangkan *padding* memberi jarak inner

elemen atau jarak antara konten dari pinggir elemen. Nilai dari margin dan padding dapat berupa satuan px atau %. Contoh:

```
<html>
   <head>
     <style>
       div {
        margin: 10px;
        padding: 10px;
        background-color: lightblue;
        border: 1px solid black;
      }
     </style>
   </head>
   <body>

       <div>
           This div element has a top margin of
           100px, a right margin of 150px, a
           bottom margin of 100px, and a left
           margin of 80px.
       </div>

   </body>
</html>
```

Tampilan:



- Posisi Relatif dan Absolut

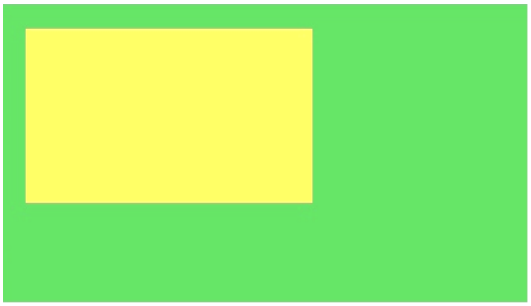
  Untuk menjelaskan perbedaan Relative dan Absolut kita langsung ke contoh kasus. Buatlah markup seperti berikut ini:

```
<div class="satu">
  <div class="dua"></div>
</div>
```

Tambahkan *style* berikut pada dokumen yang sama.

```
*{
margin:0;padding:0;
}
.satu{
  width:400px;
  height:200px;
  background:green;
  padding:20px;
}
.dua{
  width:200px;
  height:100px;
  background:orange;
  padding:20px;
}
```

Tampilan:



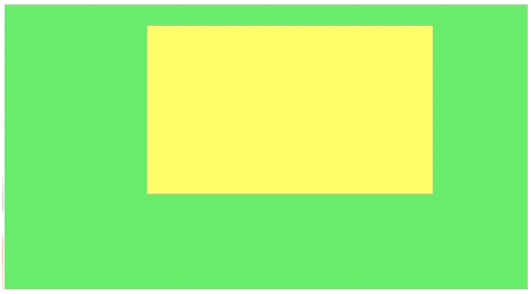
Setiap element secara *default* memiliki *position static*, ketika position bernilai *static*, properti *Top, Bottom, Left*, atau *Right* tidak dapat digunakan. Sebagai contoh, tambahkan properti *left*:100px pada elemen dengan nama class .dua, maka tidak akan terjadi perubahan sama sekali.

```
  .dua{
    width:200px;
    height:100px;
    background:orange;
    padding:20px;
    left:100px;
  }
```

Untuk itu kita perlu memberi *position relative*, dengan begitu elemen class dua akan bergeser 100px dari kiri terhadap elemen *parent*-nya.

```
.dua{
  width:200px;
  height:100px;
  background:orange;
  padding:20px;
  left:100px;
  position:relative;
}
```

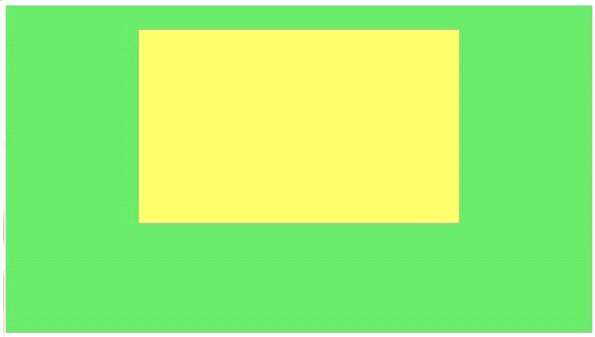
Tampilan:



Dengan menambahkan *relative position,* posisi suatu elemen dapat ditentukan. Sementara itu, *Absolute position* akan mengeluarkan elemen dari *parent*-nya dan mencari elemen parent terdekat yang memiliki *position:relative*, sehingga 100px-nya elemen dua tidak lagi dihitung terhadap elemen satu, melainkan dari elemen *body* (browser).

```
    .dua{
           width:200px;
           height:100px;
           background:orange;
           padding:20px;
           left:100px;
           position:absolute;
    }
```

Tampilan:



Kemudian tambahkan properti *bottom* dengan nilai 0, maka elemen dua akan menempel pada bagian bawah browser, karena kita menentukan jarak dari bawah dengan 0px.

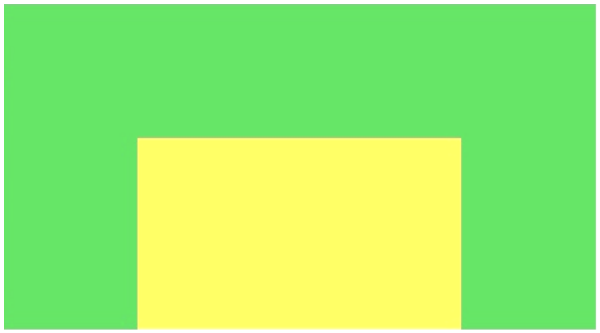
```
.dua{
      width:200px;
      height:100px;
      background:orange;
      padding:20px;
      left:100px;
      bottom:0;
      position:absolute;
}
```

Tampilan:

Agar *Absolute Position* dapat kembali ke dalam elemen satu, tambahkan *position:relative* pada elemen satu, maka elemen dua akan kembali berada pada elemen.satu.

```
   .satu{
     width:400px;
     height:200px;
     background:green;
     padding:20px;
     position:relative;
   }
```

Tampilan:



- Display

  Untuk menjelaskan jenis display, buatlah markup seperti berikut ini:

```
<!DOCTYPE html>
<html>
<head>
<style>
   p {color: red;}

   p.ex1 {display: none;}
   p.ex2 {display: inline;}
   p.ex3 {display: block;}
   p.ex4 {display: inline-block;}
</style>
</head>
<body>
   <h1>The display Property</h1>
   <h2>display: none:</h2>
   <div>
```

```
    Lorem ipsum dolor sit amet, consectetur
    adipiscing elit. Etiam semper diam at erat
    pulvinar, at pulvinar felis blandit. <p
    class="ex1">HELLO    WORLD!</p>    Vestibulum
    volutpat tellus diam, consequat gravida libero
    rhoncus ut.
    </div>

    <h2>display: inline:</h2>
    <div>
      Lorem ipsum dolor sit amet, consectetur
      adipiscing elit. Etiam semper diam at erat
      pulvinar, at pulvinar felis blandit. <p
      class="ex2">HELLO   WORLD!</p>   Vestibulum
      volutpat tellus diam, consequat gravida
      libero rhoncus ut.
    </div>

    <h2>display: block:</h2>
    <div>
      Lorem ipsum dolor sit amet, consectetur
      adipiscing elit. Etiam semper diam at erat
      pulvinar, at pulvinar felis blandit. <p
      class="ex3">HELLO   WORLD!</p>   Vestibulum
      volutpat tellus diam, consequat gravida
      libero rhoncus ut.
    </div>

    <h2>display: inline-block:</h2>
    <div>
      Lorem ipsum dolor sit amet, consectetur
      adipiscing elit. Etiam semper diam at erat
      pulvinar, at pulvinar felis blandit. <p
      class="ex4">HELLO   WORLD!</p>   Vestibulum
      volutpat tellus diam, consequat gravida
      libero rhoncus ut.
    </div>

</body>
</html>
```

Tampilan:

Tampak jelas, makna inline berarti sebuah bagian berada sebaris, sedangkan makna block berarti bagian tersebut memiliki jarak antar blok.

5.3. Rangkuman

Cascading Style Sheets (CSS) merupakan cara untuk memberikan design atau *style* pada sebuah halaman web. Terdapat tiga cara penggunaan CSS yaitu Inline CSS, Internal

CSS dan External CSS. Properti *style* pada CSS yang sering digunakan yaitu *color, background, height, width, margin dan padding.*

5.4. Tugas Latihan

1. Siapkan *file* CSS berikut, kemudian simpan dengan nama *style*.css di dalam folder assets

```
body {
    font-family: sans-serif;
    width:750px;
    margin:0 auto;
}
header {
    margin-top: 10px;
}
nav {
    margin-top: 10px;
}
article {
    margin-top: 10px;
    border-top: 1px solid #ccc;
}
h1 {
    font-family: serif;
    margin-bottom: 0;
}
.image-wrapper {
    width: 100px;
}
.image-wrapper img {
    width: 100%;
}
.image-wrapper span {
    font-family: sans-serif;
    font-size: 10px;
    color: #ccc;
}
.image-wrapper-detail {
    width: 300px;
}
.image-wrapper-detail img {
```

```
    width: 100%;
}
p {
    font-size: 14px;
}
footer{
    margin-top: 30px;
    border-top: 1px solid #ccc;
    padding: 5px;
    text-align: center;
}
```

2. Buatlah layout sebuah halaman menggunakan css di atas
3. Tambahkan halaman profil sekolah dengan menambahkan *style* CSS agar halaman dapat terlihat semenarik mungkin
4. Tambahkan HTML dan CSS untuk membuat numbering berikut:

1. first
- **1.1.** first 1
  - **1.1.1.** first 1.1
  - **1.1.2.** first 1.2
- **1.2.** first 2

2. second
- **2.1.** second 1
- **2.2.** second 2

3. third

5.5. Pustaka

1. Lewis, J., Morcovits M., Advanced CSS, New York, 2009
2. Nesterkin, D. A., Functional CSS Dynamic HTML without Javascript VOLUME 3, 2015
3. https://www.w3schools.com/cssref/

BAB 6
Javascript

Javascript merupakan bahahasa pemograman untuk membuat halaman web menjadi interaktif. Javascript melengkapi HTML dan CSS dalam sebuah halaman web. Jika HTML mendefinisikan struktur web, CSS mendefinisikan layout dan *style*, maka Javascript mendefinisikan interaksi halaman web dengan *user*.

Setelah menyelesaikan Bab ini, Anda diharapkan dapat:

1. Memahami cara-cara menggunakan Javascript
2. Memadukan Javascript dengan HTML dan CSS
3. Memahami pemrograman Javascript
4. Membuat halaman web interaktif
5. Memahami penggunaan *library* JQuery

6.1. Pengantar Materi

Pada Bab 2, telah dijelaskan bahwa Javascript berperan pada *behaviour layer*. Javascript memiliki peran penting dalam membuat halaman web interaktif. Beberapa dari kemampuan Javascript ialah dalam mengubah isi *tag* dan nilai Atribut HTML maupun mengubah *selector* dan properti CSS agar web terlihat interaktif. Javascript merupakan skrip yang berjalan di sisi *client*.

Javascript juga memiliki peran sebagai otak halaman web karena kemampuannya dalam melakukan pemrograman seperti melakukan operasi matematika, mengolah struktur data, serta menerapkan struktur kendali percabangan dan perulangan. Pemrograman dibahas di Bab 7.

6.2. Isi Materi

6.2.1. Penggunaan Javascript

Javascript dapat digunakan dengan tiga metode yaitu JS in <body>, JS in <head> atau External JS. Masing-masing metode dapat dijelaskan sebagai berikut:

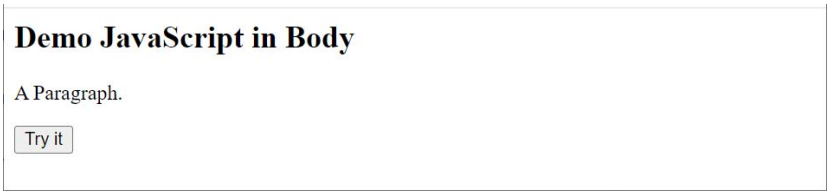
1. JS diletakkan di bagian akhir tag <body> atau <html>

   Javascript dapat diposisikan di bagian akhir <body> atau <html> dengan terlebih dahulu membuka tag <*script*> kemudian ditutup dengan </*script*>. Contohnya:

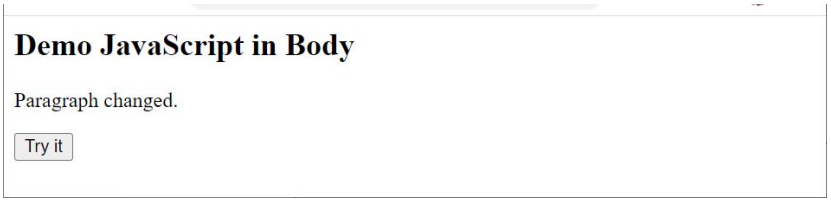
```
<html>
  <body>

    <h2>Demo Javascript in Body</h2>
    <p id="demo">A Paragraph.</p>
    <button type="button"
    onclick="myFunction()">Try it</button>
    <script>
      function myFunction() {
        document.getElementById("demo").innerHTM
        L= "Paragraph changed.";
      }
    </script>

</body>

</html>
```

*) myFunction adalah fungsi buatan yang dijelaskan lebih rinci bagian fungsi buatan javascript.

Tampilan:



Ketika tombol diklik, tampilan berubah menjadi:

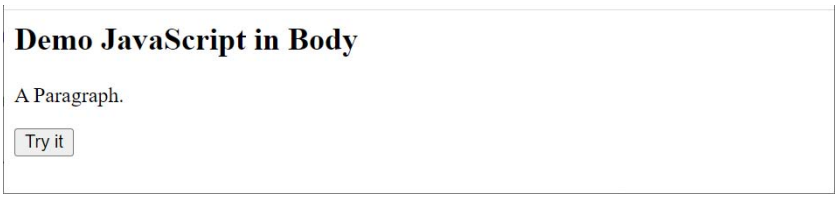


2. JS diletakkan di dalam *tag*<head>

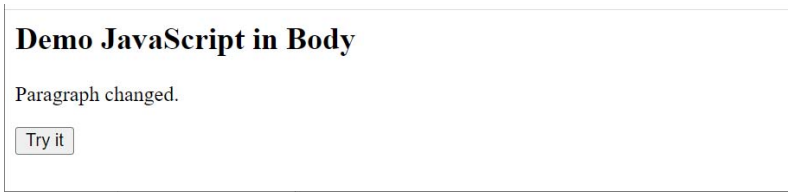
   Javascript dapat di posisikan dalam <head> dengan terlebih dahulu membuka *tag* <*script*> kemudian ditutup dengan </*script*>. Contohnya:

```
<html>
  <head>
    <script>
     function myFunction() {
      document.getElementById("demo").innerHTML
      = "Paragraph changed.";
     }
    </script>
  </head>
  <body>
    <h2>Demo Javascript in Body</h2>
    <p id="demo">A Paragraph.</p>
    <button type="button"
    onclick="myFunction()">Try it</button>
  </body>
</html>
```

Tampilan:



Ketika tombol di klik maka akan berubah menjadi:



3. External JS

External JS merupakan metode penggunaan Javascript dengan menempatkan JS pada *file* terpisah dari halaman HTML. *File* JS tersebut kemudian di panggil dengan menggunaan tag <*script*> dengan atribut src=" yang mengarah pada *file* JS. Contoh:

File script.js:

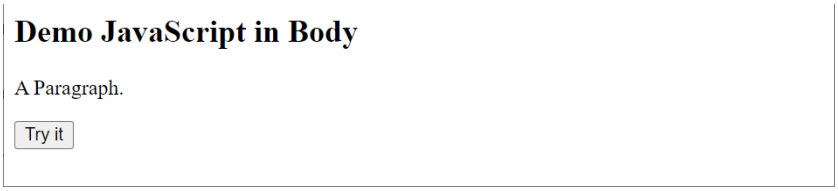
```
function myFunction() {
    document.getElementById("demo").innerHTML=
    "Paragraph changed.";
}
```

File index.html:

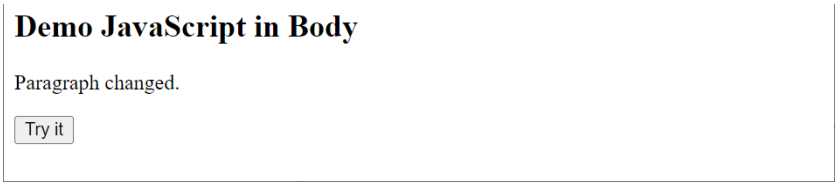
```
<html>
  <head>
    <script src="script.js"></script>
  </head>
  <body>
```

```
    <h2>Demo Javascript in Body</h2>

    <p id="demo">A Paragraph.</p>

    <button type="button"
    onclick="myFunction()"> Try it </button>

  </body>
</html>
```

Tampilan:



Ketika tombol diklik, tampilan berubah menjadi:



6.2.2. Pemrograman Javascript

Javascript mengadaptasi pemrograman berorientasi *object*. Sebuah objek tersusun atas properti dan metode (*method*) serta penanganan kejadian (*event handler*). Tanda titik (.) digunakan untuk mengakses properti, *method*, ataupun *event handler* sebuah objek. Ada tiga kelompok objek *built-in* yang dimiliki, yaitu:

- *Browser Object Model*: membuat model untuk tab browser atau *window*.

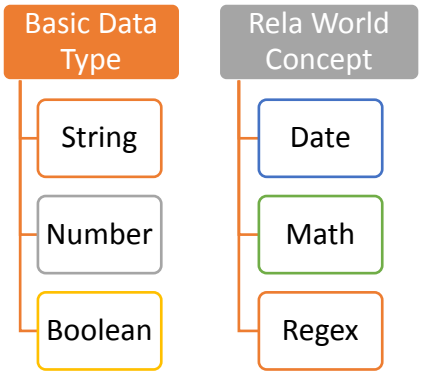
Contoh: `window.print()` dan `window.screen.width`.

Sebuah window terbagi lagi menjadi beberapa bagian:

- *Document Object Model*: membuat model untuk halaman web.

  Contoh: `document.getElementById('one')` dan `document.lastModified`

- *Global* Javascript *Object:* berupa kumpulan objek yang berhubungan satu sama lain. Objek ini diawali dengan huruf kapital.



Contoh: `str.toUpperCase()` dan `Math.PI()`

6.2.3. Document Object Model (DOM)

Javascript dapat mengakses elemen HTML dengan menggunakan Document *Object* Model (DOM). DOM merupakan susuran object dokumen HTML. Dengan memanfaatkan DOM ini, Javascript dapat melakukan manipulasi-manipulasi terkait konten, atribut dan tampilan sebuah halaman web. Javascript dapat mengakses elemen HTML dengan fungsi getElement pada tabel 6.1.

Tabel 6. 1 Akses Elemen HTML Dengan Javascript

Metode	**Deskripsi**
getElementById(id)	Memilih elemen berdasarkan nama Id
getElementsByTagName(name)	Memilih elemen berdasarkan tag HTML
getElementsByClassName(name)	Memilih elemen berdasarkan nama Class

Masing-masing metode dapat dijelaskan sebagai berikut:

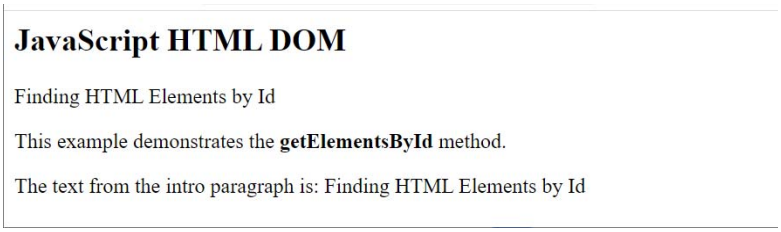
1. getElementById(id)

   Contoh:

```
<html>
  <body>
     <h2>Javascript HTML DOM</h2>
     <p id="intro">Finding HTML Elements by
     Id</p>
     <p>This example demonstrates the
```

```
    <b>getElementsById</b> method.</p>

    <p id="demo"></p>
  </body>

  <script>
      const element =
      document.getElementById("intro");
      document.getElementById("demo").innerHTML
      = "The text from the intro paragraph is:
      " + element.innerHTML;
  </script>

</html>
```

Tampilan:

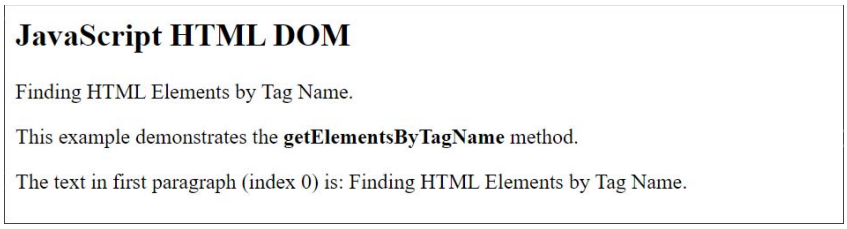


2. getElementsByTagName(name)

Contoh:

```
<html>
  <body>
     <h2>Javascript HTML DOM</h2>
     <p>Finding HTML Elements by Tag Name.</p>
     <p>This example demonstrates the
     <b>getElementsByTagName</b> method.</p>
     <p id="demo"></p>
  </body>
  <script>
     const element =
     document.getElementsByTagName("p");
     document.getElementById("demo").innerHTML
     = 'The text in first paragraph (index 0)
     is: ' + element[0].innerHTML;
  </script>
</html>
```

Tampilan:

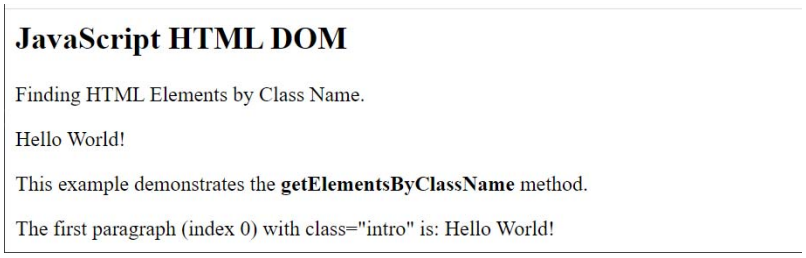


3. getElementsByClassName(name)

Contoh:

```
<html>

  <body>
     <h2>Javascript HTML DOM</h2>
     <p>Finding HTML Elements by Class Name.</p>
     <p class="intro">Hello World!</p>
     <p class="intro">This example demonstrates
     the <b>getElementsByClassName</b> method.</p>
     <p id="demo"></p>
  </body>

  <script>
     const x =
     document.getElementsByClassName("intro");
     document.getElementById("demo").innerHTML =
     'The first paragraph (index 0) with
     class="intro" is: ' + x[0].innerHTML;
  </script>

</html>
```

Tampilan:



6.2.4. Metode Menampilkan Data

Javascript dapat menampilkan data dengan beberapa cara yaitu:

- Menyisipkan elemen HTML dan atau teks dalam elemen HTML menggunakan innerHTML

  Contoh:

```
<html>
 <body>

   <h2>My First Web Page</h2>
   <p>My First Paragraph.</p>
   <p id="demo"></p>

   <script>
      document.getElementById("demo").innerHTML
      = 5 + 6;
   </script>

 </body>
</html>
```

Tampilan:

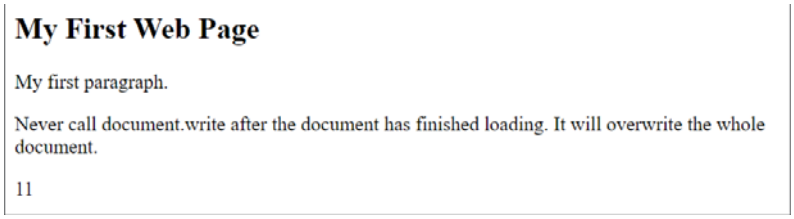
- Menampilkan elemen HTML dan atau teks menggunakan document.write()

  Contoh:

```
<html>
  <body>

     <h2>My First Web Page</h2>
```

```
    <p>My first paragraph.</p>

    <p>Never call document.write after the
    document has finished loading.
    It will overwrite the whole document.</p>

    <script>
       document.write(5 + 6);
    </script>

  </body>
</html>
```

Tampilan:



- Menampilkan alert box menggunakan window.alert()

  Contoh:

```
<html>
      <body>

      <h2>My First Web Page</h2>
      <p>My first paragraph.</p>

      <script>
         window.alert(5 + 6);
      </script>

      </body>
</html>
```

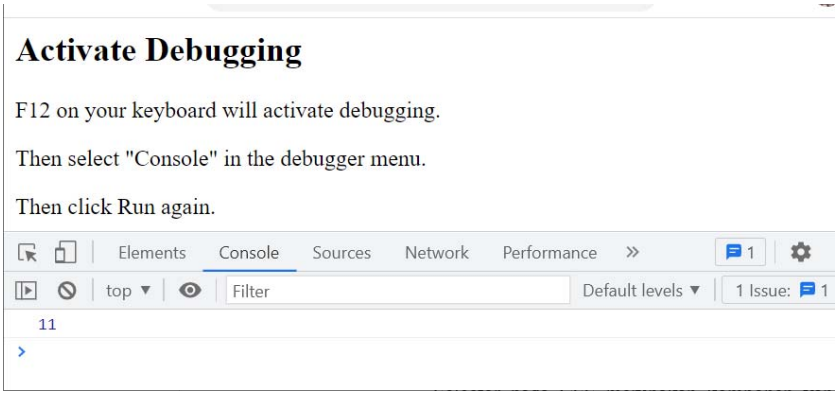
Tampilan:

- Menampilkan data dalam *console* browser menggunakan console.log(), biasanya untuk keperluan debugging.

  Contoh:

```
<html>
  <body>

     <h2>Activate Debugging</h2>

     <p>F12 on your keyboard will activate
     debugging.</p>
     <p>Then select "Console" in the debugger
     menu.</p>
     <p>Then click Run again.</p>

     <script>
        console.log(5 + 6);
     </script>

  </body>
</html>
```

Tampilan:



6.2.5. Metode Meminta Data

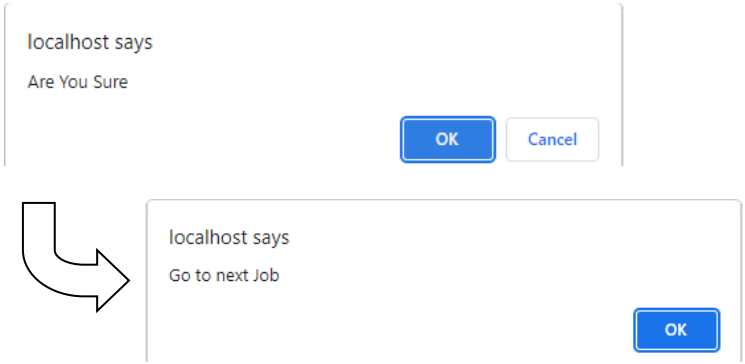
Javascript dapat meminta data dengan beberapa cara yaitu:

- Meminta persetujuan

  Contoh:

```
<script>
  var x = window.confirm("Are You Sure");
  if(x)
    window.alert("Go to next Job");
  else
    window.alert("Repeat agaia");
</script>
```

Tampilan:

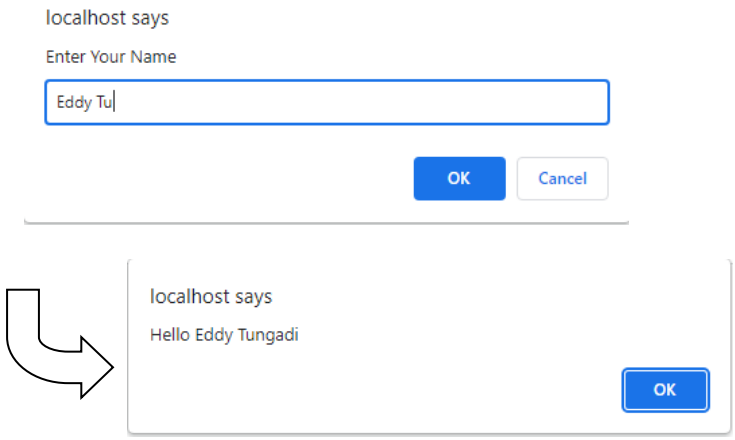


- Meminta inputan melalui window

Contoh:

```
<script>
  var n = window.prompt ("Enter Your Name");
  if(n)
    window.alert("Hello "+ n);
</script>
```

Tampilan:



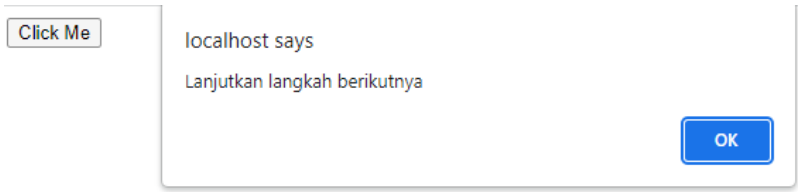
6.2.6. *Event Handler*

Event handler adalah kemampuan Javascript mendeteksi *event* (kejadian) di dalam halaman web kemudian melakukan proses tertentu. *Event handler* pernah dibahas pada tabel 3.4 dan selalu diawali dengan kata "on".

Contoh:

```
<script>
   function inform(){
      alert("Lanjutkan langkah berikutnya")
   }
</script>
<form>
    <input type="button" name="test"
    value="Click Me" onclick="inform()">
</form>
```

Tampilan:



Contoh di atas dapat pula dituliskan dalam bahasa Javascript menjadi:

```
<script>
   function inform(){
      alert("Lanjutkan langkah berikutnya")
   }
   var el = document.getElementById("test");
   el.onclick = inform;
</script>
<form>
    <input type="button" name="test"
    value="Click Me" id="test">
</form>
```

6.2.7. Metode Objek *String*

Metode pada *string* sangat dibutuhkan untuk melakukan manipulasi *string*. Untuk menggabungkan beberapa variabel *string* dapat menggunakan symbol +, sebagai contoh:

```
Greeting = 'Hai' + 'Eddy'
```

Beberapa metode *string* yang dapat digunakan seperti tabel 6.2.

Tabel 6. 2 Metode Objek String

Metode	Deskripsi	Syntax *)	Output
chartAt()	Menyalin karakter pada indeks tertentu	`var str='Kelas Javascript' str.charAt(0)`	K
concat()	Menggabungkan dua buah string	`var str='Programming ' var konten = 'Belajar Javascript' str.concat(konten)`	Kelas *Programmer* Belajar Javascript
includes()	Memeriksa apakah sebuah string berada di dalam string lainnya.	`var str = 'Programming' var konten='Belajar Javascript' str.includes(konten)`	FALSE
endsWith()	Memeriksa string diakhiri dengan string tertentu atau tidak.	`var str = 'Kelas Javascript' var konten='pt' str.endsWith(konten)`	TRUE
indexOf()	Memberikan indeks karakter pertama yang ditemukan dari sebuah string. Bila tidak ditemukan menghasilkan	`var str = 'Kelas Javascript' var konten= 's' str.indexOf(konten)`	4

	nlai -1.		

lastIndexOf()	Berkebalikan dengan indexOf, yang ditampikan index karakter terakhir yang ditemukan.	`var str = 'Kelas Javascript'`
		`var konten= 's'`
		
		`str.lastIndexOf(konten)`
			10

repeat()	Menghasilkan duplikasi string sesuai jumlah yang diinginkan.	`var str = 'Kelas Javascript'` `str.repeat(2)`	Kelas Javascript Kelas Javascript
replace()	Mengganti string lama dengan string yang baru	`var str = 'Kelas Javascript'` `str.replace(nama,'Belajar Javascript')`	Belajar Javascript
slice()	Mengambil string dengan rentan indeks awal dan akhir.	`var str = 'Kelas Javascript'` `str.slice(0,5)`	Kelas
startsWith()	Mengecek apakah suatu string diawali dengan string tertentu	`var str = 'Kelas Javascript'` `str.startsWith('Kel')`	true
substr()	Menyalin string dari indeks awal yang diinginkan dengan panjang tertentu.	`var str = 'Kelas Javascript'` `str.substr(0,10)`	Kelas Java
substring()	Menyalin sebagian string antara dua indeks	`var str = 'Kelas Javascript'` `str.substring(0,14)`	Kelas Javascrip
toLowerCase()	Mengubah huruf menjadi huruf kecil	`var str = 'Kelas Javascript'` `str.toLowerCase()`	kelas Javascript
toUpperC	Mengubah huruf	`var str = 'Kelas Javascript'`	KELAS JAVASCRI

ase()	menjadi huruf besar (huruf kapital)	`str.toUpperCase()`	PT
trim()	Menghapus spasi kosong dibagian awal maupun akhir	`var str = 'Kelas Javascript' str.trim()`	Kelas Javascript
toString()	Menjadikan suatu objek menjadi string	`var nilai=34 nilai.toString()`	34
search()	Mencari kecocokan sebuah string dalam string lainnya. Jika ditemukan akan menghasilkan index yang dicari dan jika tidak bernilai -1	`var str = 'Kelas Javascript' str.search('s')`	4
match(pola)	Mencari berdasarkan pola yang diberikan. Jika tidak ditemukan bernilai null	`var str = 'Kelas Javascript' str.match (/K/)`	K
split()	Memotong string sesuai pembatas yang diberikan	`var str = 'Kelas Javascript' str.split(' ')`	Kelas, Javascript

*) gunakan metode pada bagian Metode Menampilkan Data untuk memnculkan hasil di browser

6.2.8. Metode Objek Numerik

Tipe data numerik memerlukan penanganan agar dapat dioperasikan. Berikut beberapa fungsi numerik dapat dilihat pada tabel 6.3.

Tabel 6. 3 Metode Numerik

Metode	Deskripsi	Syntax *)	Output
toFixed()	Melakukan pembulatan angka desimal di belakang koma (kembalian berupa string)	`let num = 5.56789;` `num.toFixed(2);`	5.57
toPrecision()	Melakukan pembulatan dengan jumlah tertentu (kembalian berupa string)	`let num = 0.001658;` `num.toPrecision(2)`	0.17
toExponential()	Mengubah nilai ke bentuk notasi exponent	`let num = 0.56789;` `num.toExponential();`	5.6789e-1
isNaN()	Mengecek sebuah nilai bukan angka (Not a Number)	`isNaN('Hello')`	True

parseInt()	Mengubah nilai string menjadi integer	`var a="123",b="456";`
		`a+b;`
		`parseInt(a)+parseInt(b);`
			`123456`
			
			`579`

parseFloat()	Mengubah nilai string menjadi desimal	`var a="12.3E-3";` `parseFloat(a);`	`0.0123`

*) gunakan fungsi pada bagian menampilkan data untuk memnculkan hasil di browser

6.2.9. Fungsi Buatan Javascript

Fungsi ini dibuat oleh *programmer* untuk memenuhi kebutuhan yang belum tercakup pada metode *built-in* Javascript. Untuk membuat fungsi gunakan *syntax* berikut:

```
function nama(parameter1, parameter2, ...) {
   // kode diletakkan di sini
}
```

Semua variabel yang diletakkan di dalam fungsi bersifat lokal yang hanya dikenali di dalam fungsi itu saja. Untuk mengembalikan hasil fungsi gunakan fungsi *return*.

Untuk memanggil fungsi, tuliskan nama fungsi serta parameter yang dibutuhkan fungsi tersebut. Sebagai contoh:

```
<script>
  function ganjilgenap(nilai_local) {
     if(nilai_local % 2 == 0)
       return "Genap";
     else
       return "Ganjil";
  }
  var nilai = 5;
  alert (nilai + " termasuk bilangan " +
  ganjilgenap(nilai));
</script>
```

6.2.10. Librari jQuery

jQuery menawarkan cara yang lebih sederhana dalam melakukan tugas Javascript, bebas dari pengkodean Javascript yang rumit, sehingga pengaplikasiannya cepat dan konsisten serta dapat berjalan pada browser apapun. Penyederhanaan lainnya antara lain:

- Elemen HTML maupun selektor CSS merupakan *selector* di jQuery.
- Akses *selector* lebih menyeluruh dan fleksibel jika dibandingkan dengan menggunakan kueri DOM.
- Performansi yang lebih baik karena metode-metode jQuery

membolehkan update hirarki DOM dan dapat menjelajahi hirarki elemen dengan satu baris perintah, sehingga tugas sulit seperti animasi elemen sangat mudah dilakukan.

- Penanganan *event* dimungkinkan melalui metode-metode yang memasukkan *event listeners* pada elemen yang diinginkan tanpa melakukan pengkodean berarti.

Untuk menggunakan *library* ini, sisipkan kode berikut pada dokumen HTML:

1. Sumber diunduh dari http://jquery.com dan dsimpan di folder root web

```
<head>
   <script src="jquery-3.6.0.min.js"></script>
</head>
```

2. Sumber disimpan di CDN (Content Delivery Network). Misalnya menggunakan CDN dari Google.

```
<head>
   <script
   src="https://ajax.googleapis.com/ajax/libs/jqu
   ery/3.6.0/jquery.min.js"></script>
</head>
```

Syntax dasar jQuery:

```
$(selector).action()
```

Tanda $ mendefinisikan akses jQuery. Sebuah *selector* melakukan "query (atau menemukan)" elemen HTML. Beberapa *selector* pada jQuery pada tabel 6.4:

Tabel 6. 4 Selector jQuery

Selector	**Merujuk**
`$(this)`	Elemen sendiri
`$("p")`	Elemen <p>
`$(".test")`	Elemen dengan nama class “test"
`$("p.test")`	Elemen <p> yang memiliki class “test”
`$("p:first")`	Elemen <p> pertama dalam dokumen
`$("#test")`	Elemen dengan id="test"
`$("tr:even")`	Elemen <tr> bernomor genap
`$("[href]")`	Elemen yang memiliiki Atribut “href”
`$("p:nth-child(2)")`	Elemen <p> pada urutan/ anak kedua dalam elemen parent-nya
`$("div > p")`	Elemen <p> yang merupakan anak langsung dari elemen
`$("div p")`	Elemen <p> yang merupakan keturunan dari elemen (tidak harus langsung)
`$("div + p")`	Elemen <p> yang bertetangga langsung dengan elemen
`$("div ~ p")`	Elemen <p> yang muncul setelah elemen
`$("input:not(:empty)")`	Elemen <input> yang tidak kosong

`$(":input"):`
`$(":text")`
`$(":checkbox")`
`$(":radio")`
`$(":file")`
`$(":submit")`
`$(":selected")`
`$(":checked")`
`$(":enabled")`
`$(":dosabled")`
	Elemen <input> :
	Ttipe text
	Tipe checkbox
	Tipe radio
	Tipe *file*
	Tipe submit
	Semua item dropdown terpilih
	Semua radio/checkbox terpilih
	Semua elemen form aktif
	Semua elemen form tidak aktif

`$("a[target!='_blank']")`	Elemen yang memiliki Atribut target bukan blank

`$("[title^='Helo']")`	Elemen dengan Atribut title diawali kata “Helo”
`$("[href$='.jpg']")`	Elemen dengan Atribut title diakhir kata “.jpg”

Sebuah *selector* memiliki *action()* yang disebut juga *method* yang akan dikerjakan pada *selector* tersebut. Beberapa manipulasi yang dapat dilakukan:

- Manipulasi elemen HTML/DOM
- Manipulasi *style* CSS
- Manipulasi event
- Manipulasi efek dan animasi
- Asyncronous Javascript and XML
- Lainnya

Masing-masing contoh method dapat dilihat pada tabel 6.5.

Tabel 6. 5 Action jQuery

Method	**Deskripsi**
HTML/CSS	
`addClass()/ removeClass()/ toggleClass()`	Menambahkan/ menghapus/ toggle kelas pada selector
`hasClass()`	Mengecek apakah selector memiliki class tertentu
`after()/ before()`	Menambahkan konten setelah/ sebelum selector dan selector diletakkan di awal
`insertAfter()/ insertBefore()`	Sama dengan after/ before, hanya parameter selector diletakkan di belakang
`empty()`	Menghapus semua konten (elemen dan teks) dari selector

`text()`	Mengambil atau mengambil semua teks di dalam selector
`html()`	Mengambil atau menambahkan semua konten di dalam selector
`val()`	Mengambil atau menambahkan nilai dari Atribut value pada elemen-elemen form
`attr("Atribut")`	Mengambil atau menambahkan nilai pada Atribut html
`css()`	Menambahkan *style* css pada selector
`prop()`	Mengambil atau menambahkan nilai pada properti *style*
`prepend()/ append()`	Menambahkan konten yang diletakkan di awal/ di akhir
`prependTo()/ appendTo()`	Sama dengan prepend/ append, hanya parameter selector diletakkan di belakang
`scrollTop()/ scrollLeft()`	Mengambil atau menambahkan posisi scrollbar vertikal/ horizontal dari selector
`height()/width()`	Mengambil atau menambahkan tinggi/ lebar dari selector
`innerHeight()/ innerWidth()`	Mengambil atau menambahkan tinggi/ lebar dari selector, termasuk padding namun border tidak
`outerHeight()/ outerWidth()`	Mengambil atau menambahkan tinggi/ lebar dari selector, termasuk padding dan border
Jelajah elemen	
`first()/ last()`	Memilih elemen pertama / terakhir yang sesuai dengan selector
`parent()/`	Memilih elemen parent / child dari

`children()`	selector
`next()/ prev()`	Memilih elemen setelah / sebelum selector
`closest()`	Memilih elemen parent pertama dari selector
`not()`	Memilih elemen yang tidak sesuai
`find()`	Memilih elemen turunan yang sesuai dari selector
`eq(n)`	Memilih selector yang sesuai dengan nomot indeks n
`slice(n)`	Mengeluarkan selector yang sesuai sebanyak n
`each(function())`	Mengeksekusi fungsi untuk tiap elemen
Event: Mouse Event	
`click()`	Event klik
`dblclick()`	Event double klik
`mouseover()/ mouseout()`	Event mouseover/ mouseout
`hover()`	Event mouseover + mouseout
`scroll()`	Event mouse discroll
`submit()`	Event form disubmit
`event.pageX()`	Membaca posisi mouse dari kiri dokumen
`event.pageY()`	Membaca posisi mouse dari atas dokumen
Event: Keyboard Event	
`change()`	Event perubahan pada elemen input
`blur()`	Event blur pada elemen input
`focus()`	Event fokus pada elemen input

`keypress()`	Event keyboard ditekan pada elemen input
Event: Lainnya	
`event.preventDefault()`	Mencegah aksi default dari event
`event.which()`	Membaca key keyboard atau mouse yang ditekan dalam kode numerik
`select()`	Memilih teks dalam elemen input
`trigger(event)`	Memicu event tertentu
`ready()`	Mendefinisikan fungsi yang dieksekusi ketika DOM selesai dipasang
Efek	
`animate(`*`style`*`)`	Menjalankan animasi dengan *style* yang diinginkan
`show()/hide()/toggle()`	Memunculkan/ menyembunyikan/ toggle elemen selector dengan beberapa level durasi
`fadeIn()/ fadeout()/ fadeToggle()`	Memunculkan/ menyembunyikan/ toggle selector secara fading dengan beberapa level durasi
`slideUp()/ slideDown()/ slideToggle()`	Memunculkan/ menyembunyikan/ toggle selector secara sliding dengan beberapa level durasi
Asyncronous Javascript and XML	
`$.ajax()`	Memanggil data dengan metode asyncrounous
`$.get("target", {object}, function(output))`	Memanggil data dari alamat target server dengan metode HTTP GET, dapat menyertakan data tambahan berupa objek dan kembalian berupa output
`$.getJSON()`	Memanggil data dalam format JSON dengan metode HTTP GET

`$.get`*`Script`*`()`	Mengambil (dan mengeksekusi) Javascript dengan metode HTTP GET
`$.post()`	Mengambil data dari server dengan metode HTTP POST
`load()`	Mengambil data dari server dan meletakannya hasilnya pada selector
`serialize()`	Menyatukan elemen-elemen form dalam format nama=nilai menjadi sepenggal string sebelum dikirim
`serializeAll()`	Menyatukan elemen-elemen form menjadi array nama dan nilainya

Berikut contoh pemanfaatan jQuery:

- Dokumen HTML Form

```
<!DOCTYPE html>
  <style>
    .row{
      display: flex; justify-content: space-between;
      margin:10px;
    }
  </style>
  <form method="post" id="form_cari">
   <div class="row">
    <label for="prodi">Pekerjaan</label>
    <select id="pekerjaan" name= "jenis_pekerjaan">
      <option value="">Pilih jenis
      pekerjaan...</option>
      <option value="pegawai">Pegawai</option>
      <option value="buruh">Buruh</option>
    </select>
   </div>
   <div class="row">
    <label for="kelas">Status Pekerjaan</label>
    <select id="status" name="status_pekerjaan">
      <option value="">Status belum
      tersedia...</option>
    </select>
   </div>
   <div class="row">
```

```
    <button id="bfilter" type="submit"> Cari
    </button>
   </div>
   <div id="form_status"></div>
  </form>
  <script src="jquery-3.5.1.min.js"></script>
  <script>
    $(document).ready(function(){
      $("#form_cari ").submit(kirim_data);
      $("#pekerjaan").change(function(){
         pekerjaan = $(this).val();
         $.post("coba.php",{pekerjaan: pekerjaan},
         function(output){
          $("#status").html(output);
          console.log(output);
         });
      });
    });
    function kirim_data(e){
      if($("#pekerjaan").val() != '' &&
      $("#status").val() != ''){
          $("#form_status").html("Data berhasil
          terkirim");
          var data = $(this).serialize();
          console.log(data);
      }else
          $("#form_status").html("Periksa kembali
          isian pekerjaan dan status");
      e.preventDefault();

</script>
```

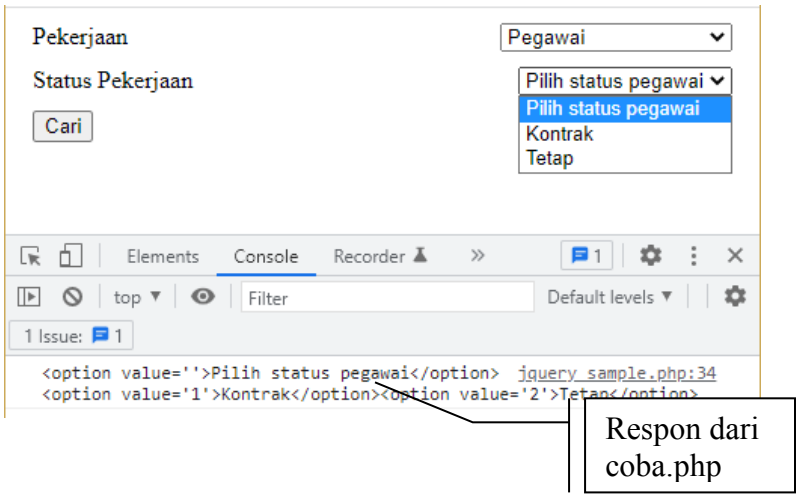
- Script coba.php sebagai script yang menerima data serta memberi respon balik yang sesuai

```
<?php
   $pekerjaan = $_POST['pekerjaan'];
   echo "<option value=''>Pilih status
   $pekerjaan</option>";
  if($pekerjaan == 'pegawai'){
     echo "<option value='1'>Kontrak</option>";
     echo "<option value='2'>Tetap</option>";
  }else if($pekerjaan == 'buruh'){
     echo "<option value='1'>Harian</option>";
     echo "<option value='2'>Proyek</option>";
  }
?>
```

Tampilan:

- HTML Form

- Script coba.php merespon perubahan dropdown pekerjaan



- Script coba.php merespon submit form yaitu melalui fungsi kirim_data()

6.3. Rangkuman

Javascript merupakan bahasa pemograman yang berfungsi untuk membuat halaman web menjadi interaktif. Javascript dapat terhubung dengan halaman web HTML dengan memanfaatkan Object Model. Javascript dapat memilih objek pada halam web berdasarkan *tag*, nama id atau nama *class*. Input dan output Javascript menggunakan beberapa metode, demikian juga untuk metode penanganan *string* dan numerik.

6.4. Tugas Latihan

1. Buatlah sebuah form yang menerima input nama dan email, kemudian tampilkan nama dan email tersebut dalam bentuk alert "Terima kasih *Nama*! Bukti akan dikirimkan ke *email* Saudara".
2. Buatlah fungsi checkEmail agar string yang diinputkan wajib memiliki karakter @ dan . memanfaatkan fungsi match.
3. Lakukan pengecekan format email menggunakan metode checkEmail kemudian buatlah sebuah paragraph di samping isian email yang menampilkan pesan Email Valid atau Invalid menggunakan metode getElement
4. Tentukan hasil dari kode berikut:

```
<script>
   var bil = 1;
   do {
     document.write(bil+ "<br>");
     bil++;
```