

Stefano Bortolato



Web server e siti web. Manuale pratico in stile How To

Manuale 2

21 giugno 2024

La scienza senza Dio è tecnologia.

La scienza con Dio è carità.

Web server e siti web.

Manuale pratico in stile How To 2

Dopo aver realizzato un web server base affrontiamo un po' di teoria per conoscere qualcosa di più su alcune delle tecnologie che abbiamo usato.

Chi è interessato ad affrontare solo la parte pratica può tranquillamente saltare al manuale 3.

In queste pagine inizieremo presentando il significato di alcuni termini che possono sembrare sinonimi perfetti, ma non lo sono. Di seguito vedremo l'argomento dei motori di ricerca per capire come segnalare un sito e cosa curare per salire nel ranking,

A seguire affronteremo il tema delle impostazioni di PHP, per passare, poi, a studiare un po' il linguaggio HTML con i CSS, le immagini, le tracce multimediali ecc...

Concluderemo l'itinerario con un capitoletto sull'SQL per capirlo, conoscerlo e usare qualche query base.

Sommario

1. Premessa.....	2
2. Tipologie e tecnologie.....	2
3. Motori di Ricerca.....	4
4. Tuning PHP.....	6
5. Teoria HTML.....	15
6. SQL: MariaDB.....	34
7. Conclusione.....	43
8. Indice.....	44

1. Premessa

In questo secondo manuale del corso concentriamo molta teoria.

L'obiettivo non è di offrire un completo percorso di formazione per ciascuna tecnologia qui presa in considerazione, ma di offrire una panoramica delle tecnologie rilevanti per creare un sito e gestire un hosting o un web server.

Pertanto tra tutte le tecnologie esistenti ci si concentrerà su quelle di uso più comune con l'ambizione di offrire un quadro di conoscenze e visioni che vada oltre al solo WordPress.

Se si desidera restare sulla parte operativa si può saldare direttamente al manuale3.

2. Tipologie e tecnologie

Iniziamo con l'introdurre alcuni concetti circa la realizzazione di siti web. Riportiamo alcuni termini simili, ma che indicano soluzioni molto diverse pur offrendo, come risultato finale, sempre un sito web.

Scopriamo e conosciamo i seguenti elementi: website builder, autoinstallazione, CMS, headless CMS, ECM, blog, wiki, e-shop – e-commerce, CDN e SEO.

- **Website builder:** si tratta di un tipo particolare di provider internet. Invece di offrire un web hosting o un VPS offrono un servizio di costruzione automatica di un sito e il mantenimento lasciando all'utente la gestione completa del sito.

Dopo aver creato l'account e aver fatto il pagamento alcune pagine ci chiedono l'aspetto, i colori, le pagine e alcune altre informazioni. In pochi minuti ci troveremo un sito completo senza aver dovuto creare un database e aver dovuto installare WordPress.

Tra i provider website builder più apprezzati ci sono Wix (<https://www.wix.com>), IONOS MyWebsite (<https://www.ionos.it/sito-web/creare-sito-web>), Hostinger (<https://www.hostinger.it>) e Squarespace (<https://it.squarespace.com>).

- **Autoinstallazione:** i Provider offrono normalmente anche il servizio di installazione automatica. Una volta entrati nel pannello di gestione del proprio hosting troviamo dei bottoni che lanciano il provisioning automatico di WordPress, Joomla, PrestaShop, ecc...

Per alcuni è un servizio incluso già nel costo dell'hosting. Altri lo offrono con qualche euro.

Il risultato finito è l'installazione del portale con una configurazione base uguale per tutti.

Web server e siti web. Manuale pratico in stile How To 2

- **CMS:** acronimo di Content Management System è una tipologia di siti web orientata a gestione contenuti generici (articoli, foto, , file per il download, servizi di form, ecc...).

Normalmente sono organizzati con un lato visibile a tutti ed un lato nascosto, a cui si accede solo dopo l'autenticazione se si è autorizzati, dove si gestiscono i contenuti ed i servizi eventualmente ospitati.

Si tratta sempre di soluzioni che hanno una certa complessità insieme a molte potenzialità.

- **HeadLess CMS:** si tratta di un CMS realizzato con un diverso approccio tecnologico. Normalmente la parte di gestione dei contenuti e dei servizi risiede sul computer di chi gestisce il sito, mentre la parte pubblica risiede sul server internet.

Con questa diversa architettura, decisamente più complessa e che richiede un po' di programmazione, si ottengono siti molto veloci e, soprattutto, estremamente sicuri.

- **ECM:** acronimo di Enterprise Content Management indica una famiglia di programmi realizzati per le aziende per gestire tutti i documenti: email, scansioni, file Word, foto, ecc...

Si tratta di soluzioni che offrono grandi potenzialità, un uso semplice per l'utente finale, ma che sono complesse, costose, interconnesse con l'IT dell'azienda, spesso installate in server (o cloud) sparsi per il mondo e interconnessi con vari servizi come, ad esempio, quelli di AI.

- **Blog:** in origine erano soluzioni per siti semplici che permettevano di pubblicare facilmente articoli e foto.

Un po' tutti i programmi che appartengono a questa famiglia sono cresciuti diventando sempre più simili ai CMS, ma mantenendo, abitualmente, una maggior facilità d'uso.

- **Wiki:** sono una tipologia di siti nati con lo scopo di permettere la pubblicazione di contenuti in modo semplice, rapido, collaborativo e partecipativo.

Un esempio di successo noto a tutti è Wikipedia (<https://www.wikipedia.org>).

- **E-shop – e-commerce:** sono una famiglia di siti che nascono per il commercio online.

Per assolvere al loro scopo normalmente prevedono e integrano tutte le funzioni logistiche e sono già predisposti per l'interoperabilità con gli operatori terzi come le banche, la fatturazione elettronica, i fornitori, ecc...

- **CDN:** le CDN (Content Delivery Network) sono delle reti speciali offerte da provider specializzati che offrono un sistema automatico per ricreare il proprio sito in più punti del mondo. In questo modo aumenta la capacità del proprio sito e si riducono sensibilmente i tempi di latenza per gli utenti più lontani.

Questi provider sono alla base del buon funzionamento dei servizi in tempo reale come, ad esempio, lo streaming TV.

Provider di questa famiglia sono, ad esempio, CloudFlare (<https://www.cloudflare.com>), CDN Gcore (<https://gcore.com>), Akamai (<https://www.akamai.com>).

- **SEO:** sigla di Search Engine Optimization. Fa riferimento ad un insieme di strategie e tecniche per migliorare la visibilità e scansione di un sito per i motori di ricerca. Il primo obiettivo del SEO è di portare il sito ai primi posti quando viene cercato tramite un motore di ricerca.

3. Motori di Ricerca

Normalmente i motori di ricerca scoprono i siti perché qualche altro sito ha dei link che puntano al sito nuovo.

Per accelerare i tempi e garantire una indicizzazione in *tempo reale* esistono servizi di segnalazione normalmente a pagamento.

I motori stessi hanno una sezione riservata ai webmaster: dopo una registrazione per accedere a quest'area il webmaster può segnalare il sito e provvedere alle specifiche attività per trustare (=mettere in relazione di fiducia) il motore di ricerca con il sito e provvedere a ottimizzare la ricerca.

In Italia il motore di ricerca più apprezzato è Google, ma non è l'unico servizio esistente nel suo campo. In altri paesi il motore più usato è un altro come, ad esempio, Yandex, Baidu, Yahoo, ecc... Pertanto è opportuno tener presente che i siti nuovi vanno segnalati a più motori se desideriamo che siano facilmente individuabili un po' in tutto il mondo.

Possiamo convergere sulle seguenti buone pratiche pensando al caso di una webmaster che cura personalmente i suoi siti.

3.1. Pool di motori di ricerca

In prima approssimazione i motori a cui segnalare un sito possono essere; Google, Bing, Yandex, Baidu, Yahoo.

Esistono anche altri motori di ricerca, ma pensando ad un posizionamento generico con questi si copre un po' tutto.

Da valutare chi vogliamo raggiungere per aggiungere o togliere motori di ricerca.

Web server e siti web. Manuale pratico in stile How To 2

Infine teniamo presente che queste realtà cambiano anche rapidamente. Pertanto è opportuno tenersi aggiornati e valutare periodicamente quale motore è più indicato per il target di riferimento di un sito.

3.2. Preparazione del sito

Esistono regole generali comuni per tutti i motori di ricerca. In particolare ricordiamo le seguenti attenzioni:

- **robot.txt:** si tratta di un file di testo da mettere nella web root del sito. Contiene le indicazioni di quali percorsi vanno scansionati dal motore di ricerca e quali no.
- **Sitemap:** si tratta di una mappa del sito generata automaticamente dal sito stesso (per quelli automatici come i CMS). Normalmente i motori di ricerca seguono la sitemap per indicizzare i contenuti;
- **Meta data:** si tratta di una descrizione di 160 caratteri massimi della pagina che viene indicizzata;
- **Meta key:** si tratta delle parole chiave per classificare la pagina e per poterla ritrovare più facilmente;
- **SEO:** acronimo di Search Engine Optimization, indica un insieme di attività per migliorare l'indicizzazione e portare il sito e/o la pagine nelle prime posizioni delle serp di ricerca;
- **H1, H2, H3, paragrafo:** i motori di ricerca apprezzano molto che le pagine che indicizzano siano aderenti agli standard HTML. In particolare che il titolo sia un paragrafo H1, che il testo sia un paragrafo (`<p>`), ecc...

I motori di ricerca condividono tutti un insieme di regole comuni per *leggere* i siti, ma hanno algoritmi propri che tengono segreti, con cui apprezzano di più o meno un determinato sito o una certa pagina.

Scoprire come ragionano i motori di ricerca ed ottimizzare i siti e le pagine per renderle più facilmente ricercabili è l'attività nota come SEO.

Infine dobbiamo dire che costantemente cambiano gli algoritmi dei motori di ricerca e che l'avvento dell'intelligenza artificiale sta cambiando profondamente l'universo dei motori di ricerca che stanno passando da sistemi di scansione di siti e pagine e sistemi che *comprendono* il contenuto.

3.3. Dal motore di ricerca

Una volta preparato il nostro sito per essere sottoposto al motore di ricerca va fatta una seconda operazione di segnalazione al motore di ricerca il quale chiede delle attività per mettere in fiducia il sito segnalato.

Questa operazione nasce da due logiche:

- il motore di ricerca ha bisogno di fidelizzare sia i siti che gli utenti;
- i motori di ricerca adottano una logica zero-trust: il sito segnalato è in bassissima fiducia fino a che non dimostra di essere proprio quello che è stato segnalato.

Normalmente il tutto viene gestito con il seguente meccanismo:

- il webmaster si iscrive nel sistema del motore di ricerca, segnala il sito e indica l'URL della sitemap;
- il motore di ricerca restituisce un file o un codice da mettere tra i file del sito e nelle pagine del sito.

Fuori del controllo di tutti ci sono degli algoritmi di verifica con cui il motore di ricerca fa una serie di verifiche al fine di adempiere alle normative nazionali e di omettere contenuti compromettenti, illegali, immorali, ecc...

Pertanto i motori di ricerca omettono l'indicizzazione di siti che vendono droghe, che hanno contenuti proibiti, ecc... Ma ognuno lo fa a modo suo.

Infine teniamo presente che il criterio base con cui i motori di ricerca promuovono o declassano i contenuti, le pagine ed i siti soprattutto in base alla popolarità e all'apprezzamento degli utenti.

4. Tuning PHP

Nel manuale 1 abbiamo proceduto all'installazione di PHP in una forma base lenta e senza alcune importanti configurazioni.

Va fatto subito presente che alcune impostazioni sono specifiche solo per alcuni siti. Pertanto è saggio (o doveroso) vedere i requisiti specifici del singolo prodotto che intendiamo usare.

Dunque vediamo l'architettura di PHP, i settaggi assolutamente da fare, il file `.htaccess` e, in fine, attiviamo PHP-FPM al posto del modulo base PHP.

4.1. Architettura

PHP può essere configurato e settato da più punti.

Fondamentalmente declina la doppia logica di una stratificazione dei permessi da un controllo centrale che può delegare tutto o qualcosa agli altri livelli.

Questa complessità, difficile da comunicare in una sola frase, deriva da un'architettura pensata per un server, dove l'amministratore deve poter tenere il controllo, dove sono ospitati più siti realizzati con applicativi diversi che impiegano configurazioni diverse e dove può essere necessario autorizzare un parametro custom in una sola directory in un sito particolare.

Insieme a questo scenario dobbiamo aggiungere un'altra complicazione. Infatti PHP può essere impiegato come linguaggio nel web server, ma può essere anche un linguaggio usato da riga di comando in background, fuori dalle pagine web (è il caso, ad esempio, di processi cron o la CLI di WordPress o di NextCloud).

Detto ciò PHP mette tutte le configurazioni centrali nel file `php.ini` accessibile e modificabile solo all'amministratore del server.

Il `php.ini` lo troviamo in tre path: `/etc/php/8.1/apache2` , `/etc/php/8.1/cli` e `/etc/php/8.1/cgi`:

- `/etc/php/8.1/apache2` : sono i settaggi adottati quando PHP viene invocato come linguaggio dal software del sito (es. WordPress o Joomla);
- `/etc/php/8.1/cli` : sono i settaggi adottati quando PHP viene invocato da CLI come nel caso di processi CRON o dei web services come `wp-cli` di WordPress od `occ` di NextCloud;
- `/etc/php/8.1/cgi` : sono i settaggi adottati quando PHP viene invocato come script CGI.

A valle di questi primi settaggi, gestibili esclusivamente dall'amministratore, abbiamo la possibilità di impostare dei settaggi tramite le configurazioni del web server Apache, ovvero il file `example.com.conf` che abbiamo creato nel manuale 1. Se abbiamo creato un certificato con CertBot, come descritto nel manuale 1, il file è `001-example.com-le-ssl.conf`. In questa fase quando l'utente invoca una pagina web, Apache passa il codice sorgente a PHP (se è codice PHP) filtrando il codice stesso e, in modo limitato, ridimensionando alcuni parametri di runtime di PHP. Tramite Apache possiamo passare parametri diversi in base alla directory-file invocato e possiamo definire, con una granularità di dettaglio, cosa può controllare l'utente che ha il permesso di scrivere nello spazio web.

Web server e siti web. Manuale pratico in stile How To 2

A valle del controllo fatto dal server Apache è possibile controllare ulteriormente il comportamento di PHP tramite il file `.htaccess` scritto all'interno delle directory dello spazio web (con l'attività fatta nel manuale 1 coincide con il path `/var/www/example.com/html`). Le possibilità di manipolare le impostazioni di PHP sono limitate da quanto definito nelle configurazioni di Apache e, prima ancora, dalle impostazioni in `php.ini`. Inoltre le impostazioni in `.htaccess` si applicano solo alla directory in cui si trova e alle eventuali sotto-directory. Ad esempio:

```
html/a/.htaccess
html/a/a1
html/a/a2
html/b
html/b/b1
html/b/b2
```

le impostazioni di `.htaccess` si applicano solo ai file PHP contenuti nella directory `"a"` e alle sottodirectory `"a1"` e `"a2"`, ma non al ramo `"b"`.

Questa architettura richiede un po' di studio per apprenderla, ma risponde a una logica lineare che permette di mantenere un controllo centrale, ma di dare anche spazio di modifiche all'utente con diritti di scrittura sul server o alle applicazioni stesse.

4.2. Settaggi

All'installazione PHP ha un'impostazione molto limitata, con dimensioni piccole per l'upload di file, piccola RAM e inibite le autorizzazioni di efficacia per `.htaccess`.

Tra i parametri da impostare ci sono le direttive:

- `memory_limit` : imposta la quantità massima di RAM usabile da uno script. Solitamente è fissata a 128MB;
- `post_max_size` : indica la dimensione massima di un pacchetto inviato da un client. Per pacchetto possiamo avere un file (operazione di upload) o i dati inviati tramite un form;
- `sys_temp_dir` : directory dove vengono messi file temporanei creati da PHP. Di default non è impostata e PHP usa solo la RAM;
- `upload_max_filesize` : dimensione massima accettata per file inviati in upload. Di default è fissata a 2MB;
- `date.timezone` : imposta il fuso orario di riferimento di PHP. Di default non è impostato, ma per svariate ragioni è opportuno impostarlo settandolo sul fuso orario dove si trova il server. L'elenco delle timezone supportate da PHP è disponibile all'URL <https://www.php.net/manual/en/timezones.php> ;
- `apc.enabled` : si tratta di una cache "Alternative PHP Cache" di default presente in PHP e settata con dei valori base.

Web server e siti web. Manuale pratico in stile How To 2

Passiamo a imposta e attivare queste impostazioni. Teniamo presente che il `php.ini` di apache ha impostazioni di default molto più limitate rispetto al `php.ini` di CGI o di CLI. Questo perché il rischio ed il bisogno di sicurezza delle pagine web esposte da Apache è molto più acuto di CGI e di CLI. Pertanto le configurazioni sono diverse in ciascun scenario d'uso.

Prima di iniziare l'aggiornamento torniamo al lavoro fatto nel manuale 1 con l'installazione di PHP. Assicuriamoci di avere sulla webroot del dominio `example.com` il file `/var/www/example.com/html/info.php` con il contenuto

```
<?php
    phpinfo();
?>
```

e tramite il nostro web browser accediamo all'URL `http://www.example.com/info.php`.

Suggeriamo di tenere aperta questa pagina mentre si procede con l'aggiornamento.

Iniziamo l'aggiornamento delle impostazioni di PHP editando `/etc/php/8.1/apache2/php.ini`

```
sudo nano /etc/php/8.1/apache2/php.ini
```

```
[ ... ]
memory_limit = 256M
[ ... ]
post_max_size = 64M
[ ... ]
sys_temp_dir = "/tmp"
[ ... ]
upload_max_filesize = 63MB
[ ... ]
date.timezone = Europe/Rome
[ ... ]
[apc]
;
; Stringhe non presenti da aggiungere alla fine di php.ini
; configurazione custom di APCu
;
apc.enabled=1
apc.shm_segments=1
apc.shm_size=32M
apc.entries_hint=4096
apc.ttl=1024
apc.gc_ttl=3600
apc.mmap_file_mask=
apc.slam_defense=1
apc.enable_cli=1
apc.use_request_time=1
apc.serializer=php
apc.coredump_unmap=0
```

Web server e siti web. Manuale pratico in stile How To 2

```
; apc.preload_path=
```

Aggiorniamo `/etc/php/8.1/cli/php.ini`

```
sudo nano /etc/php/8.1/cli/php.ini
```

```
[ ... ]
post_max_size = 64M
[ ... ]
sys_temp_dir = "/tmp"
[ ... ]
upload_max_filesize = 63MB
[ ... ]
date.timezone = Europe/Rome
[ ... ]
[apc]
;
; Stringhe non presenti da aggiungere alla fine di php.ini
; configurazione custom di APCu
;
apc.enabled=1
apc.shm_segments=1
apc.shm_size=32M
apc.entries_hint=4096
apc.ttl=1024
apc.gc_ttl=3600
apc.mmap_file_mask=
apc.slam_defense=1
apc.enable_cli=1
apc.use_request_time=1
apc.serializer=php
apc.coredump_unmap=0
; apc.preload_path=
```

Aggiorniamo `/etc/php/8.1/cgi/php.ini`

```
sudo nano /etc/php/8.1/cgi/php.ini
```

```
[ ... ]
memory_limit = 256M
[ ... ]
post_max_size = 64M
[ ... ]
sys_temp_dir = "/tmp"
[ ... ]
upload_max_filesize = 63MB
[ ... ]
date.timezone = Europe/Rome
[ ... ]
[apc]
;
```

Web server e siti web. Manuale pratico in stile How To 2

```
; Stringhe non presenti da aggiungere alla fine di php.ini
; configurazione custom di APCu
;
apc.enabled=1
apc.shm_segments=1
apc.shm_size=32M
apc.entries_hint=4096
apc.ttl=1024
apc.gc_ttl=3600
apc.mmap_file_mask=
apc.slam_defense=1
apc.enable_cli=1
apc.use_request_time=1
apc.serializer=php
apc.coredump_unmap=0
; apc.preload_path=
```

Dopo aver apportato questi aggiornamenti dobbiamo attivarlo. Possiamo farlo comunicando ad Apache di ricaricare le configurazioni. In questo caso non abbiamo interruzioni di servizio, ma con alcune configurazioni avanzate o particolari potremmo non vedere gli effetti. In particolare il ricaricamento delle configurazioni non azzerà le eventuali cache precedentemente create e dovremmo attendere il timeout di scadenza delle stesse.

Il comando da dare è

```
sudo systemctl reload apache2
```

Oppure possiamo riavviare completamente Apache. Questo azzerà tutto (cache comprese), ma introduce un'interruzione di servizio e l'operazione richiede qualche secondo in più.

Il comando da dare è

```
sudo systemctl restart apache2
```

Verifichiamo gli aggiornamenti apportati.

Tramite il nostro web browser, apriamo una nuova finestra o un nuovo tab e accediamo all'URL `http://www.example.com/info.php` ; confrontiamo i valori presenti sulla finestra (o sul tab) prima dell'aggiornamento e quelli presenti dopo l'aggiornamento. Utile cercare in tutta la pagina tramite le parole chiave (ad esempio: `post_max_size`) avere un'immediata evidenza del lavoro fatto.

4.3. .htaccess

La sintassi accettata in questo file è un po' diversa da quella che abbiamo visto sopra perché parla direttamente con Apache. A seconda della direttiva viene elaborata direttamente da Apache o passata al programma responsabile per quella direttiva.

Premesso ciò tutto diventa più facile.

Per quanto riguarda PHP vedremo che le parole chiave (o parole riservate) sono quasi identiche.

Per rendere attivo questo file dobbiamo agire sulla configurazione di Apache. Riprendendo il lavoro fatto con il manuale 1 . Editiamo il file `example.com.conf` . Se avessimo attivato l'HTTPS tramite CertBot dobbiamo operare sul file `001-example.com-le-ssl.conf` . Assicuriamoci che ci sia la direttiva `AllowOverride All` per la webroot:

```
<Directory "/var/www/example.com/html">
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
</Directory>
```

Nota 1: il valore `All` rende modificabili tutte le impostazioni predefinite di PHP. Il sistema, in alternativa, può indicare una o più chiavi modificabili dal file `.htaccess` in modo da limitare le possibilità di manipolazione;

Nota 2: la direttiva `<Directory "/var/www/example.com/html">` rende attivo `.htaccess` a tutto il contesto del dominio `www.example.com` . Se, ad esempio, avessimo configurato la direttiva per la sottodirectory `"a"`

```
<Directory "/var/www/example.com/html/a">
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
</Directory>
```

l'override concesso ad `.htaccess` si applica solo alla sotto-directory `"a"` e alle eventuali ulteriori directory contenute in `"a"`. In questo caso `.htaccess` deve trovarsi dentro la directory `"a"`, ovvero nel path `/var/www/example.com/html/a` .

Normalmente il file `.htaccess` viene caricato dall'operatore che ha i diritti per usare l'SFTP per l'upload dei file.

Molti CMS o altri applicativi LAMP (es.: vTiger, PHPlist, ecc...) hanno già dei file `.htaccess` all'interno del pacchetto che impostano i corretto dimensionamento delle direttive PHP.

Ecco qualche riga di esempio tratte da un file `.htaccess` di un'installazione di una vecchia versione di Roundcube webmail (<https://roundcube.net>), un'applicazione di webmail:

Web server e siti web. Manuale pratico in stile How To 2

```
# AddDefaultCharset UTF-8
AddType text/x-component .htc

php_value upload_max_filesize 5M
php_value post_max_size 6M
php_value memory_limit 64M

php_flag session.auto_start Off
php_value session.gc_maxlifetime 21600
php_value session.gc_divisor 500
php_value session.gc_probability 1

<IfModule mod_rewrite.c>
RewriteEngine On
RewriteRule ^favicon\.ico$ skins/larry/images/favicon.ico
</IfModule>
```

4.4. PHP-FPM

Ritorniamo sul lavoro fatto nel manuale 1 dove abbiamo installato e attivato la modalità base di PHP.

Pur essendo perfettamente funzionante il modulo PHP di Apache è afflitto da una serie di limiti. In particolare è lento e introduce un sovracalcolo di rilievo all'interno di Apache.

Installiamo e attiviamo PHP-FPM (FastCGI Process Manager), presente di serie in tutte le distribuzioni Linux. Si tratta di un miglioramento del vecchio del vecchio sistema CGI. Le elaborazioni PHP vengono portate fuori Apache e parallelizzate ai calcoli fatti da Apache. In questo modo si ottengono tre grandi benefici:

- si accelera notevolmente l'elaborazione degli script PHP;
- PHP viene eseguito in una bolla esterna al web-server Apache migliorando notevolmente la sicurezza;
- come processo esterno possiamo assegnarlo ad un utente diverso dall'utente del web-server potendo creare tanti processi paralleli e indipendenti quanti sono i domini ospitati.

Si tratta del sistema default usato da Nginx, l'altro grande e diffusissimo web-server.

Fatta questa premessa per capire cosa stiamo per fare e per dare ragione della scelta fatta da tutti per mettere in produzione un web-server LAMP passiamo a dare fare l'installazione e l'attivazione. Come abbiamo fatto nel manuale 1 operiamo sempre da terminale:

- installiamo PHP-FPM

```
sudo apt install php8.1-fpm
```

- attiviamo i moduli e la configurazione di PHP-FPM

Web server e siti web. Manuale pratico in stile How To 2

```
sudo a2enmod proxy_fcgi
sudo a2enconf php8.1-fpm
```

- a questo punto abbiamo un nuovo file `php.ini` presente in `/etc/php/8.1/fpm/php.ini`. Aggiorniamo alcune direttive come fatto sopra

```
sudo nano /etc/php/8.1/fpm/php.ini
```

```
[ ... ]
memory_limit = 256M
[ ... ]
post_max_size = 64M
[ ... ]
sys_temp_dir = "/tmp"
[ ... ]
upload_max_filesize = 63MB
[ ... ]
date.timezone = Europe/Rome
[ ... ]
[apc]
;
; Stringhe non presenti da aggiungere alla fine di php.ini
; configurazione custom di APCu
;
apc.enabled=1
apc.shm_segments=1
apc.shm_size=32M
apc.entries_hint=4096
apc.ttl=1024
apc.gc_ttl=3600
apc.mmap_file_mask=
apc.slam_defense=1
apc.enable_cli=1
apc.use_request_time=1
apc.serializer=php
apc.coredump_unmap=0
; apc.preload_path=
```

- a questo punto abbiamo due processi per il web-serve: `apache2` ed il nuovo `php8.1-fpm`. Per attivare tutto dobbiamo riavviare entrambe i servizi

```
sudo systemctl restart php8.1-fpm apache2
```

- verifichiamo il corretto caricamento e funzionamento di PHP-FPM aprendo il nostro web-browser e accedendo all'URL `http://www.example.com/info.php`; troveremo diverse righe come la seguente `/etc/php/8.1/fpm`.

4.5. Composer

Composer non è propriamente una configurazione di PHP, ma un componente aggiuntivo.

Nel manuale 1 e nei paragrafi precedenti abbiamo spiegato che nel tempo PHP si è evoluto diventando progressivamente più complesso. Molto hanno contribuito i vari framework creati dai programmatori. Anche varie soluzioni di componenti esterni hanno contribuito ulteriormente a rendere più complesso il panorama. Va anche aggiunto, per completezza, che alcuni elementi funzionano correttamente solo se le altre parti sono di una certa versione.

Tutto questo ha contribuito alla creazione di un componente dedicato che usa una tecnologia consolidata nata in altri campi (in particolare Maven e Docker Composer). In pratica si crea un semplice file di testo dove ogni riga riporta un componente software da installare-prepara. In pratica un file descrittivo steso con una precisa sintassi. A questo punto sarà sufficiente, dalla riga di comando, invocare composer indicandogli quale file descrittore deve interpretare ed il software farà tutto il lavoro al posto nostro.

Alcuni software PHP, tra cui anche dei CMS, hanno iniziato a usare questa tecnologia per l'installazione. È il caso del potente e apprezzato CMS Drupal e di altri.

È opportuno essere informati dell'esistenza di questa tecnologia e di averla presente perché PHP Composer abitualmente non è nei repository e va aggiunto manualmente nel server.

5. Teoria HTML

Dopo aver creato il nostro primo server abbiamo un ambiente completo che ci permette un po' di sperimentazione e di apprendimento.

Nel capitolo precedente abbiamo imparato a migliorare la configurazione di PHP, a gestire la granulosità che offre e ad accelerarlo notevolmente.

Passiamo ora a scoprire cos'è l'HTML e a imparare qualche tag della sua sintassi. Scopriremo anche i CSS e capiremo a cosa servono. Dedicheremo un capitoletto alle immagini, uno ai video e uno all'audio. Infine dedichiamo un capitoletto anche ai JavaScript.

Il tutto non con l'obiettivo di imparare e padroneggiare queste tecnologie, ma di imparare cosa sono, come funzionano e di possedere quell'alfabeto base utile e necessario in ogni situazione ai webmaster e a chi aspira acquisire skill equivalenti.

Prima di procedere si tenga presente che tutto quello che verrà esposto è riproducibile tramite l'uso di un editor di testi semplici (indichiamo Sublime Text <https://www.sublimetext.com>) ed un web browser (indichiamo l'uso di FireFox <https://www.mozilla.org/it/firefox> per le funzioni di analisi che offre).

Web server e siti web. Manuale pratico in stile How To 2

I file html li possiamo provare o pubblicandoli sul web-server creato nel manuale 1 oppure mettendoli in una cartella locale e aprendoli con il web browser.

5.1. HTML

L'HTML è un linguaggio estremamente diffuso, standardizzato, stabile e presente in tantissime tecnologie, compresi i dispositivi di domotica.

In pratica server per codificare una qualsiasi pagina che abbia testo, immagini, bottoni, audio, video, ecc... in un formato intermedio che permetta a qualsiasi dispositivo di riasssemblare tutto per renderlo leggibile e fruibile al destinatario indipendentemente dal dispositivo che sta usando. Con questa premessa la pagina `esempio.html` potrà essere ricomposta e accessibile sia che abbiamo un computer, un tablet, uno smartphone, una smart TV, lo schermo della domotica, il cruscotto OLED dell'auto, ecc...

HTML sta per "HyperText Markup Language". Nasce ufficialmente nel 1991 con l'obiettivo di offrire uno strumento semplice per creare testi corredati da immagini e grafici con le classiche formattazioni base (corsivo, sottolineato, grassetto, ecc...).

Per raggiungere l'obiettivo il creatore si basò su una tecnica ben nota tra i programmatori: marcare, tramite un tag, i paragrafi o i caratteri.

Facendo un esempio elementare il codice

```
</h1>Titolo</h1>
<p>Testo in <b>grassetto</b>, <i>corsivo</i>, <u>sottolineato</u>.</p>
<p>Esempio HTML.</p>
```

letto da un web browser diventa come segue

Titolo

Testo in **grassetto**, *corsivo*, sottolineato.

Esempio HTML.

Figura 1: Esempio

In questo caso il significato dei tag è piuttosto evidente:

- `<h1>` : titolo di primo livello;
- `<p>` : paragrafo;
- `` : grassetto;
- `<i>` : corsivo;
- `<u>` : sottolineato.

Web server e siti web. Manuale pratico in stile How To 2

Quando l'HTML fu inventato si provvide ad un set limitato di tag che provvedevano a tutte le formattazioni standard per i testi. Progressivamente il set è stato esteso per soddisfare alle nuove esigenze che, passo dopo passo, emergevano. Attualmente l'HTML è alla versione 5 e incorpora, tra le cose più interessanti, la riproduzione dell'audio, del video e animazioni di base.

Ogni tag ha un comportamento e un aspetto grafico predefinito, ma modificabile tramite i CSS che vedremo dopo. Se non vogliamo pagine elaborate abbiamo già tutto senza dover aggiungere nulla.

I tag che vale la pena conoscere sono i seguenti:

- `<!--...-->` : commento. Tutto quello compreso tra `<!--` e `-->` viene ignorato;
- `<a>` : hyperlink. Rende una parola, una frase o un oggetto un link;
- `<audio>` : inserisce un oggetto audio con tanto di controlli per il play;
- `` : mette il testo in grassetto;
- `
` : interruzione di riga. Introduce un ritorno a capo senza terminare il paragrafo;
- `<button>` : inserisce un bottone;
- `` : testo barrato;
- `` : testo enfaticizzato (corsivo);
- `<h1>` `<h2>` `<h3>` `<h4>` `<h5>` `<h6>` : titolo di livello 1, 2, 3, 4, 5, 6;
- `<hr>` : linea di separazione orizzontale;
- `<i>` : testo in corsivo;
- `` : inserisce un'immagine;
- `<ins>` : testo evidenziato come nuovo (sottolineato);
- `` : voce di un elenco. Vedere `` e ``;
- `<mark>` : testo evidenziato in giallo;
- `` : elenco numerato. Ogni elemento è compreso tra i tag `` e ``;
- `<p>` : paragrafo di testo;
- `<pre>` : paragrafo preformattato;
- `<small>` : testo piccolo;
- `` : testo in grassetto;
- `<sub>` : carattere o testo in pedice;
- `<sup>` : carattere o testo in apice;

Web server e siti web. Manuale pratico in stile How To 2

- `<video>`: inserisce un oggetto video con tanto di controlli per il play;
- ``: elenco puntato. Ogni elemento è compreso tra i tag `` e ``.

Per un completo e corretto codice HTML dobbiamo incapsulare il testo con i tag tra un blocco dichiarativo di apertura e uno di chiusura:

```
<!DOCTYPE html>
<html>
<body>

codice HTML

</body>
</html>
```

quindi il codice esempio creato all'inizio per essere completo deve essere come segue:

```
<!DOCTYPE html>
<html>
<body>

<h1>Titolo</h1>
<p>Testo in <b>grassetto</b>, <i>corsivo</i>, <u>sottolineato</u>.</p>
<p>Esempio HTML.</p>

</body>
</html>
```

A questo punto possiamo passare a sperimentare l'uso diretto dell'HTML:

- con Sublime Text creiamo il file `esercizio.html`;
- popoliamo il file con del codice HTML;
- mettiamo il file nel web-server o in una cartella locale;
- accediamo al file con il web-browser.

Per spiegazioni, completamenti, esempi e test online suggeriamo di usare l'ottima piattaforma W3schools - HTML Tutorial (<https://www.w3schools.com/html/default.asp>).

Un'attività che suggeriamo con forza per capire come funziona l'HTML e come vengono realizzate alcune pagine è di spiare il codice della pagina. Quindi, quando siamo sulla pagina web che ci interessa e la stiamo osservando tramite Firefox premiamo da tastiera la combinazione CTRL+U oppure "Menu → Altri strumenti → Sorgente pagina".

5.2. CSS

Nel capitoletto precedente abbiamo visto come, tramite pochi tag HTML, possiamo efficacemente creare una pagina formattata.

Però se volessimo che il titolo di livello 1 (tag `<h1>`) fosse rosso invece che nero non è possibile con l'HTML base.

Web server e siti web. Manuale pratico in stile How To 2

Però possiamo modificare lo stile del titolo tramite i CSS per ottenere che i titoli di primo livello diventino rossi.

I CSS, acronimo di Cascading Style Sheets, è lo strumento con cui possiamo modificare le impostazioni di default degli stili predefiniti e crearne di nuovi.

I CSS sono strumenti molto potenti, ma di conseguenza hanno anche una grande complessità.

Pertanto qui riportiamo una panoramica con un esempio pratico. Per un auto-apprendimento rimandiamo all'ottima risorsa W3schools CSS Tutorial (<https://www.w3schools.com/css>)

5.2.1. CSS e SCSS

Iniziamo con un altro contributo teorico. Se usiamo i CMS molto rapidamente incontreremo gli SCSS gestendo i template e le impostazioni estetiche.

Gli SCSS (Sassy Cascading Style Sheets, o Sassy CSS) sono un superset di CSS: contengono tutte le funzionalità di un CSS, ma sono ampliati per includere tutte le funzionalità. Normalmente sono file grandi.

Per accelerare l'accesso alle pagine normalmente vengono ottimizzati e compressi.

Per aumentare la flessibilità e il riuso esistono librerie pronte all'uso. Nel caso di template le troviamo all'interno del pacchetto in codice sorgente e vengono ottimizzati e compressi quando confermiamo il settaggio del template.

Molti template, inoltre, offrono pagine dove è possibile modificare sia gli SCSS, sia i CSS.

5.2.2. Inserire i CSS in una pagina HTML

Abbiamo due possibilità per inserire un CSS in una pagina HTML:

- possiamo inserire direttamente il codice nella testa del file HTML;
- oppure possiamo creare un file esterno e mettere un link auto-caricante nella testa del file HTML.

Precedentemente abbiamo parlato dell'incapsulamento del codice HTML all'interno di una dichiarazione di inizio file e una di fine file.

In realtà, in una corretta pagina HTML ha un'ampia sezione di testa, normalmente invisibile, dove troviamo meta informazioni, il caricamento dei javascript, di file CSS, ecc...

normalmente i file CSS sono esterni al file HTML. In questo mo possiamo condividere il CSS con tutte le pagine HTML del sito rendendole esteticamente omogenee,

qui vediamo come alterare il CSS del titolo di primo livello per renderlo rosso:

- tramite Sublime Text creiamo il file `esercizio02.html` ;
- popoliamo con il seguente codice

Web server e siti web. Manuale pratico in stile How To 2

```

<!DOCTYPE html>
<head>
<style>
h1 {
    color: red;
    text-align: center;
}
</style>
</head>
<html>
<body>

<h1>Titolo</h1>
<p>Testo in <b>grassetto</b>, <i>corsivo</i>, <u>sottolineato</u>.</p>
<p>Esempio HTML.</p>

</body>
</html>

```

- dopo averlo salvato apriamolo con un web-browser;
- vedremo qualcosa di uguale o molto simile all'immagine qui di seguito

Titolo

Testo in **grassetto**, *corsivo*, sottolineato.

Esempio HTML.

Figura 2: Esempio 2

5.2.3. Creare custom.css

Come detto sopra non è conveniente inserire il codice CSS dentro la pagine HTML. È preferibile creare un file esterno e le pagine html nell'header riportano un link autocaricante;

- con Sublime Text creiamo il file `custom.css` e salviamolo nella stessa directory dove si trova il file `esercizio02.html`;
- inseriamo il seguente codice in `custom.css`

```

h1 {
    color: red;
    text-align: center;
}

```

- creiamo il file `esercizio03.html` nella stessa directory dove si trovano `esercizio02.html` e `custom.css`;
- inseriamo il seguente codice

Web server e siti web. Manuale pratico in stile How To 2

```
<!DOCTYPE html>
<head>
<link rel="stylesheet" href="./custom.css">
</head>
<html>
<body>

<h1>Titolo</h1>
<p>Testo in <b>grassetto</b>, <i>corsivo</i>, <u>sottolineato</u>.</p>
<p>Esempio HTML.</p>
<a href="./esercizio02.html">Link a <code>esercizio01.html</code></a>

</body>
</html>
```

- apriamo il file `esercizio03.html` con il nostro web browser.

5.2.4. Test del file CSS

Facciamo un piccolo passo di avanzamento ulteriore. Vogliamo cambiare colore e allineamento del titolo senza modificare il file HTML:

- apriamo il file `custom.css` con Sublime Text;
- aggiorniamo il file come segue

```
h1 {
  color: blue;
  text-align: right;
}
```

- ricarichiamo dal nostro web browser il file `esercizio03.html` ;
- vedremo qualcosa di uguale o molto simile all'immagine qui di seguito

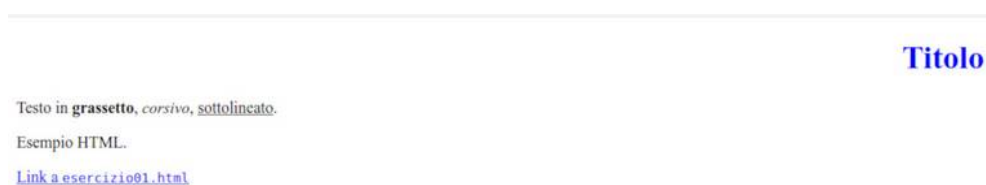


Figura 3: Esempio 3

5.3. Immagini

L'argomento immagini comporta un po' di teoria di grafica.

Iniziamo dai concetti di DPI e PPI, poi vediamo la questione delle dimensioni in PX e, in fine, il tema dei formati. Quindi vediamo come inserire l'immagine nel nostro file `esercizio04.html`.

5.3.1. DPI e PPI

Il DPI (Dots Per Inch) indica la qualità di un'immagine stampata indicando quanti punti di inchiostro sono presenti sulla carta per pollice.

Il PPI (Pixel Per Inch) indica il numero di pixel (punti luminosi) per pollice che creano l'immagine e, ovviamente, si riferisce ai monitor.

Mentre per la carta è possibile aumentare il numero di punti di inchiostro presenti per pollice (basta fare punti di inchiostro più piccoli), con i monitor i pixel sono fissi. Se il costruttore ha messo 100 pixel per pollice non possiamo in nessun modo aumentarne il numero. Questa differenza intrinseca nei supporti (carta e monitor) portano a dover gestire in modo diverse le immagini:

- se vanno su stampa le misuriamo in DPI. Possiamo aumentarli a patto di avere una stampante sufficientemente precisa;
- se vanno su monitor (cellulare, schermo computer, schermo TV, ecc...) le misuriamo in PPI per lo schermo ha 100 PPI e se l'immagine ha una qualità superiore, tutti i pixel che eccedono vengono buttati.

In merito ai monitor, malgrado le tecnologia stiano cambiando, di fatto vengono visualizzati ancora 72 pixel per pollice. Aggiungiamo la considerazione che nel web è importante tenere piccole le dimensioni degli oggetti. Se aggiungiamo che pixel maggiori significa file più grandi il risultato diventa matematico: nel web le immagini si misurano in PPI, la qualità è a 72 PPI e vanno tagliate alla dimensione con cui vengono visualizzate nella pagina web.

Aggiungiamo un ultimo dettaglio: diversi motori per siti web (come i CMS, i blog system, ecc...) tagliano e ridimensionano automaticamente le immagini quando vengono caricate alleggerendo il lavoro del web editor.

5.3.2. PX

Se la qualità delle immagini digitali si misura in DPI, per la stampa, e in PPI per l'uso su monitor, la dimensione (altezza e larghezza) si misura in centimetri (o millimetri, o pollici, o...) per la stampa, e solo in pixel per la visualizzazione su monitor.

Per la carta da molti anni esistono precisi standard di dimensione, per i monitor è un po' più complicata la situazione e ancora fluida. Inoltre se per i monitor degli smartphone si guarda alla dimensione altezza e larghezza espressa in pixel, per i monitor e le televisioni si indica la diagonale in pollici, mentre per conoscere la dimensione dobbiamo scoprire se è un monitor standard, HD, full HD, 4K, ecc...

Senza andare oltre crediamo che sia evidente che ci sono molte variabili e poche costanti. Pertanto possiamo trovare un riferimento d'oro nei seguenti criteri:

- tagliare le immagini alla misura in pixel che devono essere viste;

Web server e siti web. Manuale pratico in stile How To 2

- per o cellulari prendiamo il taglio più diffuso e tagliamo le immagini mai oltre alla dimensione del cellulare;
- per i tablet applichiamo lo stesso criterio (quindi ci serve una seconda immagine);
- per i monitor applichiamo lo stesso criterio (quindi ci serve una terza immagine).

Anche in questo caso molti motori di siti provvedono ad un taglio e ridimensionamento automatico. Tramite la tecnologia adaptive dei template riconoscono il monitor e procedono ad un ridimensionamento automatico oltre alla ricombinazione degli oggetti che compongono la pagina.

5.3.3. JPG, PNG, WEBP, AVIF, SVG

Altro argomento da conoscere è il formato delle immagini. Il cambio di formato non è un semplice cambio di estensione del file, ma significa un cambio degli algoritmi con cui vengono codificate e decodificate. Esistono molti modi di codificare le immagini, ma sostanzialmente, se vogliamo file di piccole dimensioni dobbiamo per forza ridurre l'informazione contenuta nell'immagine facendo decadere la qualità dell'immagine stessa. Non è raro vedere, magari alla TV, immagini sgranate con evidenti quadrati. Di solito si codifica questo effetto come "effetto pixel". Si tratta del risultato matematico della riduzione dell'informazione contenuta nell'immagine che toglie particolari seguendo una progressione matematica. Quando l'immagine è piccola l'occhio non coglie le mancanze, quando l'immagine diventa grande si vedono le pixelature e i salti cromatici nelle sfumature.

Illustrato questo elemento teorico aggiungiamo che i file delle immagini devono essere il più piccole possibili.

Quanto sopra restringe il campo a pochi formati compliant:

- JPG, PNG, WEBP, AVIF: si tratta di formati bitmap (ovvero fotografici);
- PNG: è l'unico che formato che non prevede la perdita di informazione. Inoltre supporta la trasparenza a scapito, spesso, di file più grandi;
- JPG, WEBP, AVIF: nati per minimizzare le dimensioni dei file sono tutti costruiti su algoritmi a perdita d'informazione, ma usano algoritmi di diversa generazione con risultati qualitativi diversi;
- SVG: è l'unico formato vettoriale. Oltre a permettere il salvataggio di disegni permette di introdurre anche animazione. Generalmente produce file di piccole dimensioni, ma possono rallentare i browser se composti da molti elementi vanificando le piccole dimensioni.

In conclusione usiamo solo questi formati. JPG va sempre bene. SVG è ideale per disegni e piccole animazioni. Per il migliore compromesso qualità-dimensione file AVIF; ricordiamoci però che molti schermi hanno rese cromatiche e qualitative scadenti.

5.3.4. Inserire un'immagine in una pagina

Finalmente arriviamo alla parte pratica e vediamo come inserire un'immagine nel file HTML. Prepariamoci con:

- creiamo una copia del file `esercizio04.html` con nome `esercizio05.html`;
- nella directory dove si trova il file `esercizio05.html` mettiamo le immagini `esercizio05.jpg`, `esercizio05.avif`, `esercizio05.png`, `esercizio05.webp` ed `esercizio05.svg` scaricabili da
<https://www.digitalsdb.it/moodle/course/section.php?id=45>;
- con Sublime Text apriamo il nostro file `esercizio05.html`.

Il tag che permette l'inserimento delle immagini è ``.

A differenza di quasi tutti gli altri tag in questo dobbiamo sempre mettere un'informazione aggiuntiva di commento, ovvero l'argomento "alt". Il motivo non è strettamente tecnico, ma di accessibilità. Immaginiamo la situazione di un ipovedente che accede ad una pagina web e si fa assistere da un software alla lettura dello schermo. Se l'immagine non è corredata di un commento o una didascalia, il programma potrà solo dire che c'è un'immagine.

La stessa logica la dobbiamo applicare anche ai motori di ricerca. Essi, infatti, non capiscono ancora le immagini (anche se l'AI è molto migliorata) e si affidano ai commenti che accompagnano le immagini. Inoltre, se le immagini hanno questi commenti descrittivi, i motori di ricerca danno un credito maggiore ai siti e alle pagine rendendole più facilmente reperibili.

Finita questa introduzione aggiorniamo il codice html del file `esercizio05.html` come segue:

```
<!DOCTYPE html>
<head>
<link rel="stylesheet" href="./custom.css">
</head>
<html>
<body>

<h1>Titolo</h1>
<p>Testo in <b>grassetto</b>, <i>corsivo</i>, <u>sottolineato</u>.</p>
<p>Esempio HTML.
<a href="./esercizio02.html">Link a <code>esercizio01.html</code></a></p>
<hr>
<h1>Gallery immagini</h1>
<p><br>
Didascalia: Immagine esercizio JPG</p>

<p><br>
Didascalia: Immagine esercizio AVIF</p>
```

Web server e siti web. Manuale pratico in stile How To 2

```
<p><br>
Didascalia: Immagine esercizio PNG</p>

<p><br>
Didascalia: Immagine esercizio WEBP</p>

<p><br>
Didascalia: Immagine esercizio SVG</p>

</body>
</html>
```

- ricarichiamo dal nostro web browser il file `esercizio05.html` ;
- vedremo qualcosa di uguale o molto simile all'immagine qui di seguito

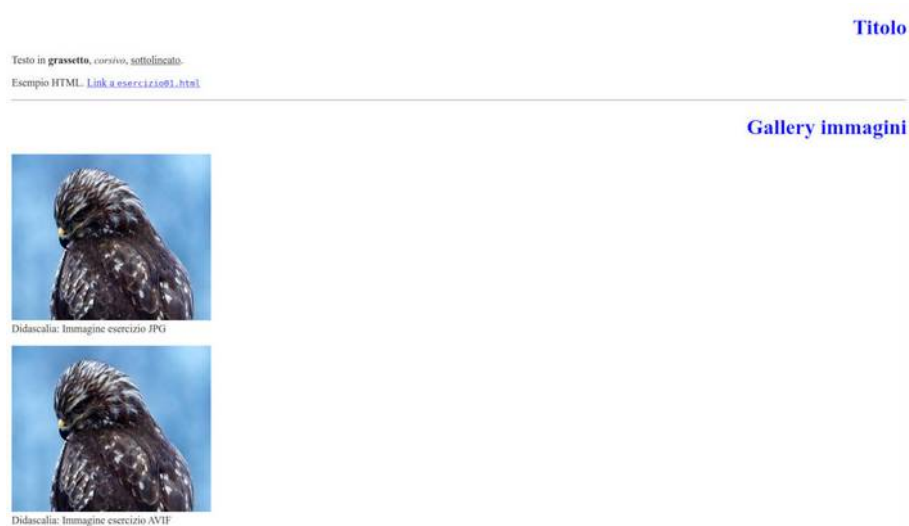


Figura 4: Esempio 5

5.4. Video

Per gli oggetti video valgono i principi teorici riportati per le immagini.

Dobbiamo aggiungere il concetto di bitrate, ovvero la quantità di dati che devono costantemente arrivare al client per poter visualizzare il video. Normalmente è il video editor che imposta tutto. Pertanto ci basta conoscere il principio.

Altro elemento che dobbiamo conoscere: il server web deve avere molta banda. Più sono gli utenti che guardano contemporaneamente il nostro video, maggiore deve essere la banda disponibile. Motivo per cui molti usano YouTube e altre piattaforme di streaming.

Un'ulteriore complicazione deriva dalle tracce audio. Anche questo elemento occupa (molto) spazio e se in streaming c'è il bitrate da garantire. Esistono diversi standard di codifica. Nel caso del file video tutto questo è invisibile, ma all'interno del nostro file video c'è un vero e proprio file nascosto che contiene tutto l'audio. Per ora non entro nel merito e ci fermiamo alla sola conoscenza del principio.

Infine è opportuno avere chiaro che i file video sono file grandi sempre.

Per i formati è in corso un'evoluzione tecnologica. Pertanto al momento, senza entrare nella discussione, teniamo come riferimento l'uso dei soli formati MP4 ed WMV.

Detto tutto ciò l'HTML 5 (quello che usiamo tutti da qualche anno) incorpora le funzioni per lo streaming ed il controllo della riproduzione degli stessi.

Procediamo a stendere un paio di righe HTML per includere un mini filmato nella nostra pagina prima delle immagini. Prepariamoci con:

- creiamo una copia del file `esercizio05.html` con nome `esercizio06.html`;
- nella directory dove si trova il file `esercizio06.html` mettiamo il video `esercizio06.mp4` scaricabili da <https://www.digitalsdb.it/moodle/course/section.php?id=45>;
- con Sublime Text apriamo il nostro file `esercizio06.html`.

Il tag che permette l'inserimento delle immagini è `<video>`:

- aggiorniamo il codice html del file `esercizio06.html` come segue:

```
<!DOCTYPE html>
<head>
<link rel="stylesheet" href="./custom.css">
</head>
<html>
<body>

<h1>Titolo</h1>
<p>Testo in <b>grassetto</b>, <i>corsivo</i>, <u>sottolineato</u>.</p>
```

Web server e siti web. Manuale pratico in stile How To 2

```

<p>Esempio HTML.
<a href="./esercizio02.html">Link a <code>esercizio01.html</code></a></p>
<hr>
<h1>Gallery video</h1>
<p><video src="./esercizio06.mp4" alt="Video esercizio 06"><br>
Didascalia: Video esercizio 06<br>
<small>Preso da <a href="https://sample-videos.com/">sample-videos.com</a></p>
<hr>
<h1>Gallery immagini</h1>
<p><br>
Didascalia: Immagine esercizio JPG</p>

<p><br>
Didascalia: Immagine esercizio AVIF</p>

<p><br>
Didascalia: Immagine esercizio PNG</p>

<p><br>
Didascalia: Immagine esercizio WEBP</p>

<p><br>
Didascalia: Immagine esercizio SVG</p>

</body>
</html>

```

- ricarichiamo dal nostro web browser il file `esercizio06.html` ;
- vedremo qualcosa di uguale o molto simile all'immagine qui di seguito

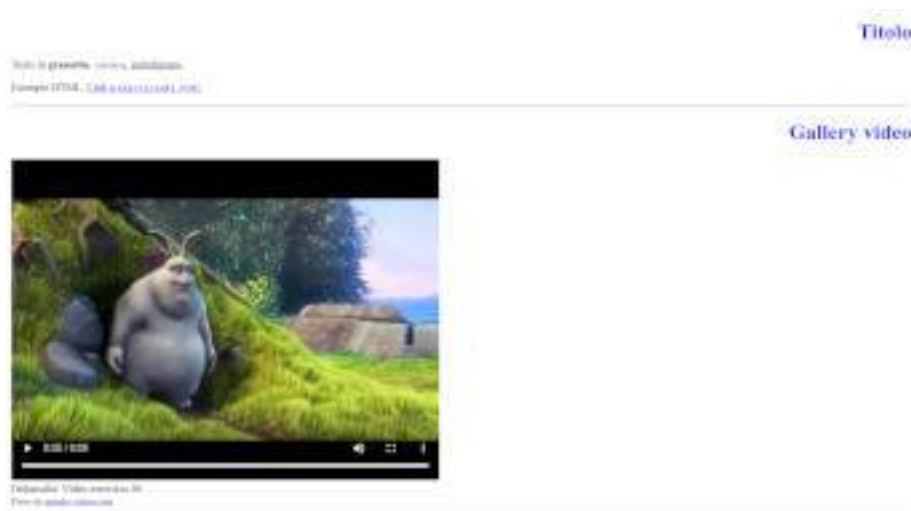


Figura 5: Esempio 6

5.5. Audio

Per gli oggetti audio valgono i principi teorici riportati per i video.

Vale il concetto del bitrate, ovvero la quantità di dati che devono costantemente arrivare al client per poter riprodurre l'audio, come spiegato per i video. Rispetto a questi, però, le tracce audio necessitano di meno informazione, pertanto abbiamo file più piccoli. Normalmente è l'audio editor che imposta tutto. Pertanto ci basta conoscere il principio come per i video.

Per i formati è in corso un'evoluzione tecnologica come per i video. Pertanto al momento, senza entrare nella discussione, teniamo come riferimento l'uso del solo formato MP3.

Detto tutto ciò l'HTML 5 (quello che usiamo tutti da qualche anno) incorpora le funzioni per lo streaming ed il controllo della riproduzione degli stessi.

Procediamo a stendere un paio di righe HTML per includere un mini file audio nella nostra pagina prima del filmato. Prepariamoci con:

- creiamo una copia del file `esercizio06.html` con nome `esercizio07.html`;
- nella directory dove si trova il file `esercizio07.html` mettiamo il video `esercizio07.mp3` scaricabili da
<https://www.digitalsb.it/moodle/course/section.php?id=45>;
- con Sublime Text apriamo il nostro file `esercizio07.html`.

Il tag che permette l'inserimento delle immagini è `<audio>`:

- aggiorniamo il codice html del file `esercizio07.html` come segue:

```
<!DOCTYPE html>
<head>
<link rel="stylesheet" href="./custom.css">
</head>
<html>
<body>

<h1>Titolo</h1>
<p>Testo in <b>grassetto</b>, <i>corsivo</i>, <u>sottolineato</u>.</p>
<p>Esempio HTML.
<a href="./esercizio02.html">Link a <code>esercizio01.html</code></a></p>

<hr>
<h1>Gallery audio</h1>
<p><audio controls>
  <source src="./esercizio07.mp3" type="audio/mpeg">
  Your browser does not support the audio element.
</audio> <br>
```

Web server e siti web. Manuale pratico in stile How To 2

```

Didascalìa: Audio esercizio 07<br>
<small>Preso da <a href="https://samplelib.com/sample-mp3.html">samplelib.com</a></p>
<hr>
<h1>Gallery video</h1>
<p><video controls>
  <source src="./esercizio06.mp4" type="video/mp4">
  Your browser does not support the video tag.
</video> <br>
Didascalìa: Video esercizio 06<br>
<small>Preso da <a href="https://sample-videos.com/">sample-videos.com</a></p>
<hr>
<h1>Gallery immagini</h1>
<p><br>
Didascalìa: Immagine esercizio JPG</p>

<p><br>
Didascalìa: Immagine esercizio AVIF</p>

<p><br>
Didascalìa: Immagine esercizio PNG</p>

<p><br>
Didascalìa: Immagine esercizio WEBP</p>

<p><br>
Didascalìa: Immagine esercizio SVG</p>

</body>
</html>

```

- ricarichiamo dal nostro web browser il file `esercizio07.html` ;
- vedremo qualcosa di uguale o molto simile all'immagine qui di seguito

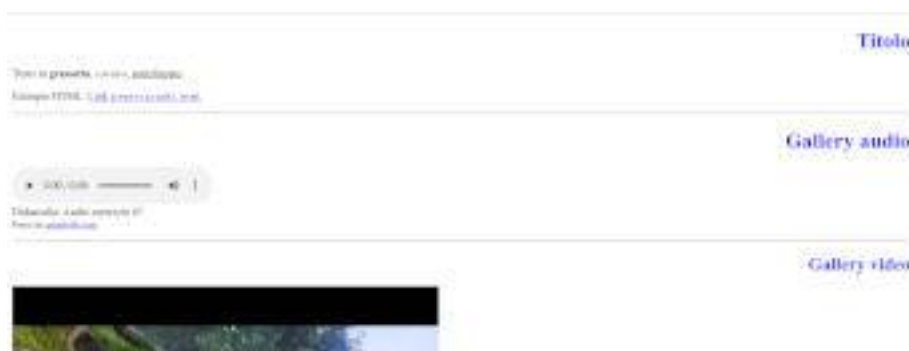


Figura 6: Esempio 7

5.6. Javascript

I javascript sono piccoli programmi inseriti all'interno di una pagina web. Scritti con un java semplificato sono concepiti per automatizzare e potenziare le pagine web. Gli sviluppatori hanno curato con grande attenzione la sicurezza. Pertanto i javascript sono molto limitati e sempre isolati all'interno del web browser.

Per impararli e capirli a fondo rimando alle ottime guide:

- W3schools - JavaScript Tutorial (<https://www.w3schools.com/js/default.asp>);
- The Modern JavaScript Tutorial (<https://javascript.info>).

Come per i CSS possiamo mettere il codice dello script all'interno della pagina web o metterlo come file esterno.

Creiamo un codice javascript "Hello, world" dimostrativo inserendo il codice javascript nel blocco head della pagina HTML. Prima del blocco audio mettiamo un blocco con un bottone "Click" che intercetterà il click del mouse e richiamerà lo script che induce il browser ad aprire una popup di messaggio con il testo "Hello, world".

Partiamo:

- creiamo una copia del file `esercizio07.html` con nome `esercizio08.html`;
- con Sublime Text apriamo il nostro file `esercizio08.html`;
- aggiorniamo il codice html del file `esercizio08.html` come segue:

```
<!DOCTYPE html>
<head>
<link rel="stylesheet" href="./custom.css">

<!-- JavaScript INIZIO -->
<script>
  function scriptEsempio8() {
    alert( 'Hello, world!' );
  }
</script>
<!-- JavaScript FINE -->

</head>
<html>
<body>

<h1>Titolo</h1>
<p>Testo in <b>grassetto</b>, <i>corsivo</i>, <u>sottolineato</u>.</p>
<p>Esempio HTML.
<a href="./esercizio02.html">Link a <code>esercizio01.html</code></a></p>
```


Web server e siti web. Manuale pratico in stile How To 2

```

<hr>
<h1>Gallery javascript</h1>
<p><button onclick="scriptEsempio8()">Click</button></p>

<hr>
<h1>Gallery audio</h1>
<p><audio controls>
  <source src="./esercizio07.mp3" type="audio/mpeg">
Your browser does not support the audio element.
</audio> <br>
Didascalìa: Audio esercizio 07<br>
<small>Preso da <a href="https://samplelib.com/sample-mp3.html">samplelib.com</a></p>
<hr>
<h1>Gallery video</h1>
<p><video controls>
  <source src="./esercizio06.mp4" type="video/mp4">
  Your browser does not support the video tag.
</video> <br>
Didascalìa: Video esercizio 06<br>
<small>Preso da <a href="https://sample-videos.com/">sample-videos.com</a></p>
<hr>
<h1>Gallery immagini</h1>
<p><br>
Didascalìa: Immagine esercizio JPG</p>

<p><br>
Didascalìa: Immagine esercizio AVIF</p>

<p><br>
Didascalìa: Immagine esercizio PNG</p>

<p><br>
Didascalìa: Immagine esercizio WEBP</p>

<p><br>
Didascalìa: Immagine esercizio SVG</p>

</body>
</html>

```

- carichiamo dal nostro web browser il file `esercizio08.html`;
- clicchiamo con il mouse sul bottone "Click"; vedremo qualcosa di uguale o molto simile all'immagine qui di seguito



Figura 7: Esempio 8

5.7. Console FireFox

Probabilmente per l'autoapprendimento aiuta molto poter guardare dentro una pagina web e guardare la comunicazione che avviene tra il proprio terminale ed il web server.

Questa attività è perfettamente possibile con gli strumenti standard in dotazione al PC. Per motivi pratici è meglio fare questa attività da un PC piuttosto che da un telefonino o un tablet.

Suggeriamo l'uso di FireFox, ma anche gli altri principali browser hanno gli strumenti di analisi delle pagine web e di analisi del traffico client-server. A questo riguardo rimandiamo alle pagine per gli sviluppatori della Mozilla: Firefox Developer Edition (<https://www.mozilla.org/it/firefox/developer/>).

In questo manuale abbiamo iniziato usando FireFox. Continuiamo dando le indicazioni per questo popolare browser.

5.7.1. Informazioni sulla pagina

Premendo la combinazione CTRL+I, FireFox apre una finestra che riassume le caratteristiche di una pagina web. Vengono riportate le informazioni come il titolo, la codifica, il tipo di pagina, ma riporta che l'elenco di tutti i media (immagini, disegni, blob, audio, video, ecc....) contenuti nella pagina.

Utile, come dire, per avere una panoramica tecnica della pagina web.

5.7.2. Sorgente della pagina

Premendo la combinazione CTRL+U si apre un tab dove viene riportato tutto il codice HTML che ricevuto dal web browser.

Si tratta di informazione tecnica, ma molto utile perché ci permette di vedere quale codice realizza la pagina che vediamo.

Web server e siti web. Manuale pratico in stile How To 2

Ricordiamoci che quasi tutti i siti web hanno dei motori che li generano dinamicamente. A volte il codice HTML che riceviamo oltre ad essere creato istantaneamente, riporta link, metadati e/o script virtuali.

Teniamo presente anche che alcuni framework dei motori dei siti creano codici HTML molto compatti per ridurre le dimensioni e i tempi di trasferimento. Pertanto potremmo essere di fronte a una pagina web molto bella e molto ricca. Premendo CTRL+I potremmo vedere una sola riga.

In realtà si tratta di una riga chilometrica perché vengono tolti tutti i ritorni a capo, tutti gli spazi inutili, ecc... Il risultato è dell'informazione difficilmente leggibile da noi, ma che il computer comprende molto bene.

5.7.3. Analisi

Probabilmente è lo strumento più potente ed utile quando dobbiamo scoprire la causa di un difetto.

Sulla nostra pagina web se premiamo il bottone destro del mouse si apre un piccolo menù contestuale che ha una voce "Analizza". Cliccando su questa voce in una porzione dello schermo si apre un potente strumento di analisi, visualizzazione, ispezione, ecc....



Figura 8: Apertura "strumenti di sviluppo" in FireFox

Utile lo strumento "Seleziona un elemento" che ci permette di cliccare su un oggetto dalla pagina e vedere subito a quale parte del codice afferisce.

"Console" e "Rete" ci permette di scoprire e vedere la comunicazione che avviene tra il client ed il server remoto. È facile imbattersi in pagine che apparentemente non inviano alcun dato e scoprire che inviano, invece, informazioni. Questo strumento evidenzia sempre i problemi presenti nelle pagine e/o nella comunicazione.

Con gli strumenti a disposizione è possibile vedere e accedere ai file CSS e JavaScript. Questo aiuta molto a capire come sono stati realizzati e imparare da essi.

6. SQL: MariaDB

Vediamo, in chiave pratica, come funziona un database SQL. Usiamo la nostra installazione di MariaDB e interagiamo con il database da terminale tramite il client `mariadb`.

6.1. Premessa

All'avvento delle tecnologie dei database nacquero diverse soluzioni che declinavano la tecnologia in modo diverso e, soprattutto, reciprocamente incompatibile. Finché qualcuno pensò che fosse utile standardizzare un linguaggio comune, neutro, che funzionasse con ogni database: nacque SQL.

L'SQL (Structured Query Language) fu una fantastica idea, ma è una promessa parzialmente tradita. Quasi da subito l'SQL si divide in diversi dialetti. I vari DB implementano potenzialità che non sono usabili tramite l'SQL. Le specifiche SQL, di fatto, non sono così universali come era nelle intenzioni dei padri del linguaggio....

C'è anche da dire, per completezza, che la progressiva diffusione dell'informatica, con l'aumento verticale dell'uso dei DB in soluzioni grandi e complesse ha messo a nudo importanti limiti connessi alle tecnologie SQL. Questo ha dato rapidamente origine a database noti come "no SQL": il loro scopo e la loro caratteristica è di essere veloci a scapito, però, della rinuncia della tipicizzazione dei dati e della perdita delle relazioni.

Detto tutto ciò al nostro scopo è utile capire la grammatica base dell'SQL e, alla luce di quanto sopra esposto, capire perché alcune applicazioni Usano MySQL, altre MariaDB, altre PostgreSQL, altre DB2, ecc... Non è solo per la specifica licenza d'uso e per i costi che comporta.

Aggiungiamo un altro elemento di teoria molto rilevante. Se qualcuno ha usato Access o un altro DB desktop fondamentalmente si crea un database, con un certo numero di tabelle, ma non abbiamo bisogno di autenticazioni, definizioni dell'utente che lo userà, ecc... in un server, invece, abbiamo bisogno che il software del database gestisca contemporaneamente molti database, molti utenti, permessi diversi, con la possibilità di lavorare su grandi quantità di dati, magari distribuiti su più server e con la necessità di garantire un funzionamento continuo.

Pertanto un software desktop non è adatto. Per questo sono stati progettati applicativi adatti per i server, identificati dal nome DBMS, ovvero "Data Base Management System": fondamentalmente sono un'installazione monolitica. L'amministratore gestisce gli utenti creandoli e concedendo i permessi. L'amministratore, inoltre, crea i database, li associa agli utenti (se non hanno autorità per crearli e autogestirli), gestisce le configurazioni del server per velocizzarlo e tenerlo in sicurezza, ecc....

Web server e siti web. Manuale pratico in stile How To 2

Per i DBMS che incarnano la tecnologia relazionale abbiamo il sottogruppo RDMS in cui troviamo anche gli applicativi MySQL e MariaDB.

Alla luce di questa premessa di teoria per accedere a un database gestito da un RDBMS abbiamo bisogno 4 dati:

- il server (=host) su cui si trova l'istanza DB;
- l'utente (=user) che ha i permessi di accesso al database;
- la password dell'utente del database;
- il nome del database.

Passiamo ora alla parte pratica con lo scopo di conoscere l'alfabeto base SQL e degli RDBMS.

Ultima nota: la riga di istruzioni SQL passate all'RDBMS si chiama query.

Per un approfondimento e per l'autoapprendimento rimandiamo alla smisurata guida online della MariaDB e al Tutorial SQL:

- Documentation – MariaDB.org (<https://mariadb.org/documentation>);
- MariaDB Server Documentation, 2024-05, file PDF
(<https://mariadb.org/wp-content/uploads/2024/05/MariaDBServerKnowledgeBase.pdf>);
- GeeksforGeeks | A computer science portal for geeks. SQL Tutorial
(<https://www.geeksforgeeks.org/sql-tutorial>).

6.2. Creare un DB

Sfruttiamo l'utente root che il primo amministratore del nostro RDBMS MariaDB. L'istruzione `sudo` fa in modo che i comandi siano impersonificati dall'utente root :

- accediamo al nostro server remoto, creato nel manuale 1, tramite ssh con il nostro utente webadmin;
- connettiamoci, ora, a MariaDB tramite il comando `mariadb`

```
sudo mariadb -h localhost -u root
```

- dal monitor di MariaDB diamo la query per creare in DB esercizio9

```
CREATE DATABASE esercizio9;
```

- verifichiamo la creazione del database

```
show databases;
```

a questo punto, senza uscire dal monitor di MariaDB passiamo a creare un utente con permessi globali sul database esercizio9 .

6.3. Creare un utente per il DB

Creiamo un utente, con relativa password, con permesso di collegarsi solo dall'interno del server (=localhost) e autorizzazioni globali su database.

L'utente avrà le seguenti impostazioni:

- username: esercizio9_user
- password: esercizio9_password
- host: localhost

Username e password sono didattici. Mentre può essere una buona pratica creare username che hanno come prefisso il database a cui afferiscono, la password invece deve sempre essere seria, non come questa che è solo a scopo didattico.

Procediamo: dal monitor di MariaDB diamo le tre seguenti query:

```
CREATE USER esercizio9_user@localhost IDENTIFIED BY 'esercizio9_password';  
GRANT ALL ON esercizio9.* TO esercizio9_user@localhost WITH GRANT OPTION;  
FLUSH PRIVILEGES;
```

Alcune note esplicative:

- la prima query crea l'utente con la password;
- la seconda query concede tutti i permessi all'utente esercizio9_user sul database esercizio9;
- l'ultima query rende esecutivi i nuovi permessi.

A questo punto usciamo. Procederemo usando il nuovo utente esercizio9_user. Diamo il comando di uscita:

```
quit;
```

6.4. Create la tabella “moto” e “colore”

Le query che iniziamo a usare da qui in poi cominciano a diventare un po' corpose. Può essere una buona idea scriverle dentro Sublime Text e poi copiarle nel monitor di MariaDB.

- Colleghiamoci a MariaDB con l'utente esercizio9_user

```
mariadb -h localhost -u esercizio9_user -p  
Enter password:  
Welcome to the MariaDB monitor.  Commands end with ; or \g.  
Your MariaDB connection id is 44  
Server version: 10.6.16-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

Web server e siti web. Manuale pratico in stile How To 2

```
MariaDB [(none)]>
```

- accedere al database esercizio9

NB: MariaDB e MySQL permettono che un utente abbia accesso a più database. Pertanto dopo l'accesso bisogna sempre indicare quale database vogliamo usare

```
use esercizio9;
```

- creare la tabella "colore"

```
CREATE TABLE colore
(
    id_colore INT AUTO_INCREMENT PRIMARY KEY NOT NULL,
    colore VARCHAR(50) NOT NULL,
    note TEXT
);
```

- creare la tabella "moto"

```
CREATE TABLE moto
(
    id_moto INT AUTO_INCREMENT PRIMARY KEY NOT NULL,
    marca VARCHAR(15) NOT NULL,
    modello VARCHAR(50) NOT NULL,
    colore INT,
    note TEXT,
    FOREIGN KEY (colore) REFERENCES colore(id_colore)
);
```

A questo punto abbiamo pronte le strutture base per inserire dei dati.

6.5. Caricare N entry

Caricheremo per primi i colori, poi le moto che prenderanno il colore dalla tabella colore precedentemente popolata.

Sempre operando dal monitor di MariaDB:

- diamo la query per popolare la tabella colore

```
INSERT INTO colore (colore, note) VALUES
('Nero','Bello'),
('Rosso','Molto Mello'),
('Giallo','Bellissimo');
```

- diamo la query per popolare la tabella colore

```
INSERT INTO moto (marca, modello, colore, note) VALUES
('Aprilia', 'Tuono 1100', 1, 'Nota demo 1'),
('Aprilia', 'Tuono 660', 2, 'Nota demo 2'),
('Aprilia', 'Tuono 125', 3, 'Nota demo 3'),
('Aprilia', 'Tuareg 660', 1, 'Nota demo 4'),
('Aprilia', 'SXR 50', 2, 'Nota demo 5'),
('Aprilia', 'SX 125', 3, 'Nota demo 6'),
('Aprilia', 'SR GT 200', 1, 'Nota demo 7'),
('Aprilia', 'SR GT 125', 2, 'Nota demo 8'),
```

Web server e siti web. Manuale pratico in stile How To 2

```
('Aprilia', 'RSV4', 3, 'Nota demo 9'),
('Aprilia', 'RX 125', 1, 'Nota demo 10'),
('Aprilia', 'RS 660', 2, 'Nota demo 11'),
('Aprilia', 'RS 125', 3, 'Nota demo 12');
```

A questo punto abbiamo un mini database popolato. Passiamo a fare una ricerca.

6.6. Cercare

Sempre operando dal monitor di MariaDB passiamo delle query di ricerca.

Prima di iniziare diamo note di teoria per comprendere meglio le query che seguiranno:

- **WHERE** : sostanzialmente è l'istruzione cerca;
- **=** : è l'operatore di ricerca. Vengono restituiti i record che hanno il valore identico alla chiave passata;
- **LIKE** : è l'operatore di ricerca. Semplificando: vengono restituiti i record che hanno il valore simile alla chiave passata. Supporta i wildcard;
- **%** : è un carattere wildcard. Sta per: qualsiasi carattere e qualsiasi lunghezza;
- **_** : è un carattere wildcard. Sta per: un solo carattere qualsiasi.

Per conoscenza è opportuno sapere che MariaDB supporta anche la ricerca full text.

I seguenti esempi di query di ricerca sono corredati anche del risultato che ottenete.

- Ricerca con restituzione di tutte le colonne:

```
SELECT *
FROM moto LEFT JOIN colore ON moto.colore = colore.id_colore
WHERE moto.modello = "Tuono 660";
+-----+-----+-----+-----+-----+-----+-----+-----+
| id_moto | marca  | modello | colore | note          | id_colore | colore | note          |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 2      | Aprilia | Tuono 660 | 2      | Nota demo 2   | 2        | Rosso  | Molto Mello   |
+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.001 sec)
```

- Ricerca con restituzione delle sole colonne d'interesse

```
SELECT moto.marca,moto.modello,colore.colore,moto.note
FROM moto LEFT JOIN colore ON moto.colore = colore.id_colore
WHERE moto.modello = "Tuono 660";
+-----+-----+-----+-----+
| marca  | modello | colore | note          |
+-----+-----+-----+-----+
| Aprilia | Tuono 660 | Rosso  | Nota demo 2   |
+-----+-----+-----+-----+
1 row in set (0.001 sec)
```


Web server e siti web. Manuale pratico in stile How To 2

- Ricerca con criterio wildcard: tutti i modelli 660

```
SELECT moto.marca,moto.modello,colore.colore,moto.note
FROM moto LEFT JOIN colore ON moto.colore = colore.id_colore
WHERE moto.modello LIKE "%660";
```

marca	modello	colore	note
Aprilia	Tuono 660	Rosso	Nota demo 2
Aprilia	Tuareg 660	Nero	Nota demo 4
Aprilia	RS 660	Rosso	Nota demo 11

3 rows in set (0.001 sec)

- Ricerca con criterio wildcard: tutte le versioni dei modelli Tuono

```
SELECT moto.marca,moto.modello,colore.colore,moto.note
FROM moto LEFT JOIN colore ON moto.colore = colore.id_colore
WHERE moto.modello LIKE "Tuono%";
```

marca	modello	colore	note
Aprilia	Tuono 1100	Nero	Nota demo 1
Aprilia	Tuono 660	Rosso	Nota demo 2
Aprilia	Tuono 125	Giallo	Nota demo 3

3 rows in set (0.001 sec)

- Ricerca con criterio wildcard: tutte le versioni doppi zero di ogni modelli

```
SELECT moto.marca,moto.modello,colore.colore,moto.note
FROM moto LEFT JOIN colore ON moto.colore = colore.id_colore
WHERE moto.modello LIKE "%00%";
```

marca	modello	colore	note
Aprilia	Tuono 1100	Nero	Nota demo 1
Aprilia	SR GT 200	Nero	Nota demo 7

2 rows in set (0.001 sec)

- Ricerca con criterio wildcard doppio: tutte le versioni dei modelli S come seconda lettera

```
SELECT moto.marca,moto.modello,colore.colore,moto.note
FROM moto LEFT JOIN colore ON moto.colore = colore.id_colore
WHERE moto.modello LIKE "_S %";
```

marca	modello	colore	note
Aprilia	RS 660	Rosso	Nota demo 11
Aprilia	RS 125	Giallo	Nota demo 12

2 rows in set (0.001 sec)

6.7. Modificare un record

Nel linguaggio SQL si parla di `update` di un valore.

Procediamo con due esempi: nel primo rendiamo rosso (id = 2) il modello “Tuono 125”, Nel secondo cambiamo la nota al modello “RS 125”. Riportiamo la query per l’update e subito dopo la query con l’output per vedere l’aggiornamento fatto

- cambiamo il colore al modello “Tuono 125”

```
UPDATE moto SET colore = '2' WHERE modello = 'Tuono 125';
Query OK, 0 rows affected (0.001 sec)
Rows matched: 1  Changed: 0  Warnings: 0

SELECT moto.marca,moto.modello,colore.colore,moto.note
FROM moto LEFT JOIN colore ON moto.colore = colore.id_colore
WHERE moto.modello = 'Tuono 125';
+-----+-----+-----+-----+
| marca  | modello | colore | note          |
+-----+-----+-----+-----+
| Aprilia | RS 125  | Rosso  | Nota demo 12 |
+-----+-----+-----+-----+
1 row in set (0.001 sec)
```

- cambiamo il colore al modello “RS 125”

```
UPDATE moto SET note = 'Nuova nota per RS 125' WHERE modello = 'RS 125';
Query OK, 0 rows affected (0.016 sec)
Rows matched: 1  Changed: 0  Warnings: 0

SELECT moto.marca,moto.modello,colore.colore,moto.note
FROM moto LEFT JOIN colore ON moto.colore = colore.id_colore
WHERE moto.modello = 'RS 125';
+-----+-----+-----+-----+
| marca  | modello | colore | note          |
+-----+-----+-----+-----+
| Aprilia | RS 125  | Giallo | Nuova nota per RS 125 |
+-----+-----+-----+-----+
1 row in set (0.001 sec)
```

6.8. Cancellare un record

L’ultima azione che ci serve vedere è la cancellazione di un record. L’istruzione SQL è `delete`.

Procediamo con la cancellazione del modello RSV4, poi di tutti i modelli Tuono e poi, con una sola azione, di tutti i modelli ancora presenti nella tabella moto . Operiamo sempre dal monitor di MariaDB.

Web server e siti web. Manuale pratico in stile How To 2

- Iniziamo chiedendo l'elenco completo delle nostre moto

```
SELECT moto.id_moto,moto.marca,moto.modello,colore.colore,moto.note
FROM moto LEFT JOIN colore ON moto.colore = colore.id_colore;
```

id_moto	marca	modello	colore	note
1	Aprilia	Tuono 1100	Nero	Nota demo 1
2	Aprilia	Tuono 660	Rosso	Nota demo 2
3	Aprilia	Tuono 125	Rosso	Nota demo 3
4	Aprilia	Tuareg 660	Nero	Nota demo 4
5	Aprilia	SXR 50	Rosso	Nota demo 5
6	Aprilia	SX 125	Giallo	Nota demo 6
7	Aprilia	SR GT 200	Nero	Nota demo 7
8	Aprilia	SR GT 125	Rosso	Nota demo 8
9	Aprilia	RSV4	Giallo	Nota demo 9
10	Aprilia	RX 125	Nero	Nota demo 10
11	Aprilia	RS 660	Rosso	Nota demo 11
12	Aprilia	RS 125	Giallo	Nuova nota per RS 125

12 rows in set (0.001 sec)

- diamo il comando di cancellazione del modello RSV4

```
DELETE FROM moto WHERE modello = 'RS 125';
Query OK, 1 row affected (0.016 sec)
```

- visualizziamo l'elenco aggiornato della moto

```
SELECT moto.id_moto,moto.marca,moto.modello,colore.colore,moto.note
FROM moto LEFT JOIN colore ON moto.colore = colore.id_colore;
```

id_moto	marca	modello	colore	note
1	Aprilia	Tuono 1100	Nero	Nota demo 1
2	Aprilia	Tuono 660	Rosso	Nota demo 2
3	Aprilia	Tuono 125	Rosso	Nota demo 3
4	Aprilia	Tuareg 660	Nero	Nota demo 4
5	Aprilia	SXR 50	Rosso	Nota demo 5
6	Aprilia	SX 125	Giallo	Nota demo 6
7	Aprilia	SR GT 200	Nero	Nota demo 7
8	Aprilia	SR GT 125	Rosso	Nota demo 8
9	Aprilia	RSV4	Giallo	Nota demo 9
10	Aprilia	RX 125	Nero	Nota demo 10
11	Aprilia	RS 660	Rosso	Nota demo 11

11 rows in set (0.001 sec)

- diamo il comando di cancellazione di tutti i modelli Tuono

```
DELETE FROM moto WHERE modello like 'Tuono %';
Query OK, 3 rows affected (0.017 sec)
```

Web server e siti web. Manuale pratico in stile How To 2

- visualizziamo l'elenco aggiornato della moto

```
SELECT moto.id_moto,moto.marca,moto.modello,colore.colore,moto.note
FROM moto LEFT JOIN colore ON moto.colore = colore.id_colore;
```

```
+-----+-----+-----+-----+-----+
| id_moto | marca   | modello   | colore | note           |
+-----+-----+-----+-----+-----+
|      4  | Aprilia | Tuareg 660 | Nero   | Nota demo 4   |
|      5  | Aprilia | SXR 50     | Rosso  | Nota demo 5   |
|      6  | Aprilia | SX 125     | Giallo | Nota demo 6   |
|      7  | Aprilia | SR GT 200  | Nero   | Nota demo 7   |
|      8  | Aprilia | SR GT 125  | Rosso  | Nota demo 8   |
|      9  | Aprilia | RSV4       | Giallo | Nota demo 9   |
|     10  | Aprilia | RX 125     | Nero   | Nota demo 10  |
|     11  | Aprilia | RS 660     | Rosso  | Nota demo 11  |
+-----+-----+-----+-----+-----+
8 rows in set (0.001 sec)
```

- diamo il comando di cancellazione di tutte le moto rimaste nella tabella

```
DELETE FROM moto;
```

```
Query OK, 8 rows affected (0.015 sec)
```

- visualizziamo l'elenco aggiornato della moto

```
SELECT moto.id_moto,moto.marca,moto.modello,colore.colore,moto.note
FROM moto LEFT JOIN colore ON moto.colore = colore.id_colore;
```

```
Empty set (0.001 sec)
```

Abbiamo svuotato tutta la tabella moto con una sola istruzione.

6.9. Uscita dal RDBMS

Ricordiamo che dobbiamo sempre dal monitor del nostro database, così da chiudere la connessione.

Il comando, lo abbiamo visto all'inizio, è il seguente:

```
quit;
```

7. Conclusione

Probabilmente per chi è orientato al fare questo secondo manuale è stato una delusione se si è fermato a leggerlo.

In realtà il web master ed il web design sono skill che richiedono molte conoscenze, molte di più di quello che si pensa. Tool pronti all'uso, che permettono un rapidissimo risultato possono essere un grande inganno: se un aspirante web master realizza un buon prodotto usando uno di questi pacchetti, crollerà letteralmente alla prima richiesta di una qualsiasi personalizzazione o cambio.

Questo manuale di teoria, ben lontano dall'offrire un percorso formativo che permetta di raggiungere adeguate conoscenze e competenze nelle tecnologie trattate, ha il pregio di consegnare una serie di informazioni che espandono il confine di quanto visto nel manuale 1.

L'obiettivo principale è proprio di dare questa percezione: c'è molto da imparare, anche se non intendiamo diventare programmatori del web o web designer o DBA (=Data-Base Administrator. Per il vero si parla anche di DSA: Database System Administrator).

Un'infarinatura sulle tecnologie, inoltre, è un'informazione preziosa per orientarsi e sapere cosa chiedere o dove guardare all'insorgere di un progetto, di un bisogno o di un problema.

Ci sentiamo anche di affermare, sulla base dell'esperienza personale e di diversi informatici, che poche nozioni base, come quelle presentate qui, molte volte permettono di venire a capo di problemi apparentemente irrisolvibili.

8. Indice

1. Premessa.....	2
2. Tipologie e tecnologie.....	2
3. Motori di Ricerca.....	4
3.1. Pool di motori di ricerca.....	4
3.2. Preparazione del sito.....	5
3.3. Dal motore di ricerca.....	6
4. Tuning PHP.....	6
4.1. Architettura.....	7
4.2. Settaggi.....	8
4.3. .htaccess.....	12
4.4. PHP-FPM.....	13
4.5. Composer.....	15
5. Teoria HTML.....	15
5.1. HTML.....	16
5.2. CSS.....	18
5.2.1. CSS e SCSS.....	19
5.2.2. Inserire i CSS in una pagina HTML.....	19
5.2.3. Creare custom.css.....	20
5.2.4. Test del file CSS.....	21
5.3. Immagini.....	21
5.3.1. DPI e PPI.....	22
5.3.2. PX.....	22
5.3.3. JPG, PNG, WEBP, AVIF, SVG.....	23
5.3.4. Inserire un'immagine in una pagina.....	24
5.4. Video.....	26
5.5. Audio.....	28
5.6. Javascript.....	30
5.7. Console FireFox.....	32
5.7.1. Informazioni sulla pagina.....	32
5.7.2. Sorgente della pagina.....	32
5.7.3. Analizza.....	33

Web server e siti web. Manuale pratico in stile How To 2

6. SQL: MariaDB.....	34
6.1. Premessa.....	34
6.2. Creare un DB.....	35
6.3. Creare un utente per il DB.....	36
6.4. Create la tabella "moto" e "colore"	36
6.5. Caricare N entry.....	37
6.6. Cercare.....	38
6.7. Modificare un record.....	40
6.8. Cancellare un record.....	40
6.9. Uscita dal RDBMS.....	42
7. Conclusione.....	43
8. Indice.....	44