MIME Type

El **MIME Type** (o **Tipo MIME**) és un concepte fonamental per a qualsevol desenvolupador web o PHP, ja que afecta directament com els servidors i els navegadors processen i mostren els fitxers.

Què és el MIME Type?

MIME significa **M**ultipurpose **I**nternet **M**ail **E**xtensions. Originalment es va crear per etiquetar el contingut de correus electrònics, però avui dia és l'estàndard universal utilitzat per HTTP i altres protocols per identificar la **naturalesa i el format del document** (fitxer, dada o stream) que s'està transmetent.

Un MIME Type té sempre la mateixa estructura bàsica:

Exemples comuns

Tipus Principal	Subtipus (Exemples)	Descripció
`text`	`text/html`, `text/plain`, `text/css`	Fitxers de text dissenyats per ser llegits, o amb formatació.
`image`	`image/jpeg`, `image/png`, `image/svg+xml`	Dades d'imatges.
`application`	`application/json`, `application/pdf`, `application/zip`	Dades o fitxers binaris que requereixen una aplicació específica per ser processats. Inclou la majoria dels documents i dades API.
`video`	`video/mp4`, `video/webm`	Formats de vídeo.
`audio`	`audio/mpeg`, `audio/ogg`	Formats d'àudio.

Per a què serveix en el món web?

El MIME Type serveix essencialment per dir al client (navegador) **com ha de gestionar la dada rebuda**.

1. **Interpretació del Navegador:** Quan un servidor envia una resposta HTTP, inclou la capçalera `Content-Type`. Si el navegador rep `Content-Type: text/html`,

sap que ha de renderitzar el codi com una pàgina web. Si rep `image/jpeg`, sap que l'ha de mostrar com una imatge.
2. **Seguretat:** És una defensa important contra atacs. Si el navegador rep un fitxer anomenat `script.jpg`, però la capçalera `Content-Type` és `image/jpeg`, el navegador el tractarà com una imatge, no com a codi executable.
3. **Descàrregues Forçades:** Serveix per indicar al navegador que, en lloc de mostrar el contingut, l'ha de descarregar. Això es fa sovint utilitzant un tipus genèric com **`application/octet-stream`** juntament amb la capçalera `Content-Disposition: attachment`.

MIME per un desenvolupador

Com a desenvolupador PHP, has de tenir el control sobre els MIME Types en dos àmbits:

1. Enviar arxius o dades a client (Output)

Quan utilitzes un script PHP per servir un fitxer (per exemple, una imatge o un PDF desat fora de l'arrel pública), has de definir manualment la capçalera `Content-Type` abans de la sortida de qualsevol dada. (Veure Llavors, com descarregar fitxers?)

PHP

```
// Exemple per servir un fitxer JSON:
header('Content-Type: application/json');
echo json_encode(['status' => 'ok']);

// Exemple per servir un fitxer PDF:
header('Content-Type: application/pdf');
header('Content-Disposition: inline; filename="document.pdf"');
// ... codi per llegir el fitxer i enviar-lo (readfile()).
```

2. Validació uploads (Input)

Quan els usuaris pugen fitxers, la validació del MIME Type és una capa de seguretat fonamental. **Mai s'ha de confiar en `$_FILES['arxiu']['type']`**, ja que aquest valor és subministrat pel navegador i pot ser falsificat fàcilment.

Per a una validació segura amb PHP, cal examinar el contingut real del fitxer temporal (`tmp_name`):

- **Recomanat (PHP 5.3+):** Utilitzar l'extensió **`Fileinfo`** amb la funció **`mime_content_type($tmp_name)`**. Aquesta llegeix la capçalera real del fitxer per determinar el seu tipus, independentment de l'extensió o del que digui l'usuari.
- **Per a Imatges:** Utilitzar **`getimagesize($tmp_name)`**. Aquesta funció no només et dóna les dimensions, sinó que retorna el tipus MIME real de la imatge com a segon element de l'array. Si el fitxer no és una imatge vàlida, la funció falla, oferint una validació de seguretat addicional.

Codi Insegur	Codi Segur
`if ($_FILES['file']['type'] == 'image/jpeg')`	`$tipus_real = mime_content_type($_FILES['file']['tmp_name']);`

> El MIME Type és l'eina per controlar el que el navegador fa amb les dades i una eina essencial per garantir la seguretat en la càrrega de fitxers.

Què és `mime_content_type()`?

La funció **`mime_content_type()`** a PHP és una eina fonamental per a la **seguretat** i la **correcta identificació** dels fitxers. S'utilitza per determinar el **MIME Type real** d'un fitxer examinant-ne el seu contingut, i no confiant en l'extensió del nom de l'arxiu. La funció `mime_content_type(string $filename)` retorna el tipus MIME d'un fitxer. És part de l'extensió **Fileinfo** de PHP, la qual és la solució recomanada i més segura per a la identificació de fitxers.

El seu ús principal, especialment en el context d'uploads, és oferir una **validació de seguretat forta**. Mentre que l'array `$_FILES['arxiu']['type']` es basa en el que el navegador diu (i és falsificable), `mime_content_type()` llegeix les capçaleres del fitxer binari mateix (la seva "signatura") per determinar el seu format real.

Requisits

Per poder utilitzar aquesta funció, l'extensió **`fileinfo`** ha d'estar habilitada a la configuració de PHP (normalment ho està per defecte).

Sortides i Exemples

La funció retorna una cadena de caràcters que representa el tipus MIME. La sortida sempre segueix el format estàndard `tipus/subtipus`.

Exemples de Crida i Sortida Típica

Assumim que tenim els següents fitxers:

- `arxiu.jpg`: Un fitxer d'imatge JPEG vàlid.
- `document.pdf`: Un fitxer de document PDF.

- `codi_maliciós.jpg`: Un fitxer que té l'extensió `.jpg` però conté codi PHP (una tècnica d'atac).
- `dades.json`: Un fitxer JSON.

Codi PHP	Descripció	Sortida (Valor retornat)
`mime_content_type('arxiu.jpg')`	Retorna el tipus real d'una imatge.	`'image/jpeg'`
`mime_content_type('document.pdf')`	Identifica el format del document.	`'application/pdf'`
`mime_content_type('dades.json')`	Identifica l'estructura de dades.	`'application/json'` o `'text/plain'` (depèn de la versió de PHP i la configuració)
`mime_content_type('codi_maliciós.jpg')`	**Verificació de seguretat:** Si conté text pla o codi PHP.	`'text/plain'` o `'application/x-php'`

Ús en Validació d'Uploads (Fitxers Temporals)

Quan es processa una pujada, s'ha d'aplicar `mime_content_type()` al fitxer temporal (la ruta de `tmp_name`) abans de moure'l.

PHP

```php
<?php
$fitxer_temporal = $_FILES['fitxer_pujat']['tmp_name'];
$tipus_real = mime_content_type($fitxer_temporal);

$tipus_valits = ['image/jpeg', 'image/png'];

if (!in_array($tipus_real, $tipus_valits)) {
    // Si el tipus real no és vàlid, es rebutja.
    die("Error: El tipus de fitxer '$tipus_real' no està permès.");
}

// Si arriba aquí, el tipus MIME real és correcte.
// ... Ara es pot cridar move_uploaded_file()
?>
```

Notes importants per al desenvolupador
Fiabilitat: Aquesta funció és molt més fiable que confiar en l'extensió del fitxer o en l'array `$_FILES['arxiu']['type']`.
Rendiment: Cridar la funció sobre fitxers molt grans pot consumir recursos de CPU, ja que llegeix l'inici del fitxer. Per a fitxers extremadament grans, caldria considerar límits de temps d'execució.
Gestió d'Errors: Si el fitxer no existeix, la funció retorna `false`. És important gestionar aquesta sortida per evitar errors.
Alternativa (`finfo_open`): Existeix un mètode més avançat utilitzant l'orientació a objectes amb `finfo_open()`, que pot ser més eficient si s'han de processar molts fitxers successivament dins del mateix script, ja que la informació de les definicions MIME es carrega una sola vegada. Per a ús general, `mime_content_type()` és suficient.

Què és `getimagesize()`?

La funció **`getimagesize()`** és una altra eina de PHP crucial per a la validació de fitxers pujats, especialment quan es tracta d'imatges. No només proporciona les dimensions de la imatge, sinó que també actua com a eina de **seguretat** en retornar el **MIME Type real** de la imatge. La funció `getimagesize(string $filename)` llegeix la capçalera d'un fitxer d'imatge i retorna les seves **dimensions** (amplada i alçada) i altra informació, inclòs el **MIME Type**.

En la gestió d'uploads, `getimagesize()` té un doble propòsit:

1. **Validació d'Imatge Real:** Si el fitxer no és una imatge vàlida o el seu encapçalament està corrupte, la funció **fallarà i retornarà** **`false`**. Això invalida automàticament qualsevol fitxer que intenti amagar codi executable sota una falsa extensió d'imatge.
2. **Obtenció d'Informació:** Proporciona les dades necessàries per redimensionar la imatge o per restringir les mides d'imatge que es permeten pujar.

Sortida i exemples

La funció retorna un **array** amb múltiples elements. Si la funció falla (perquè el fitxer no existeix o no és una imatge reconeguda), retorna **`false`**.

Estructura de l'Array de Sortida

ÍNDEX	Contingut	Descripció	Exemple de Valor
`0`	Amplada (width)	L'amplada de la imatge en píxels.	`800`
`1`	Alçada (height)	L'alçada de la imatge en píxels.	`600`
`2`	Tipus d'Imatge	Un codi enter que representa el tipus d'imatge (per exemple, `IMAGETYPE_JPEG`).	`2`
`3`	Atributs HTML	Una cadena que conté les dimensions en format HTML (`width="X" height="Y"`).	`width="800" height="600"`
`mime`	**MIME Type Real**	El tipus MIME oficial de la imatge (el més crucial per a la seguretat).	`'image/jpeg'`

Exemples

Suposem que un usuari puja una imatge (que resideix temporalment a `/tmp/phpabc123`).

PHP

```php
<?php
$fitxer_temporal = $_FILES['imatge_pujada']['tmp_name'];

// 1. Cridar la funció i gestionar el possible error
$info_imatge = getimagesize($fitxer_temporal);

if ($info_imatge === false) {
    // Si falla, no és una imatge vàlida o l'arxiu està corrupte.
    die("Error de seguretat: El fitxer no és una imatge reconeguda.");
}

// 2. Extreure la informació
$amplada = $info_imatge[0];
$alçada = $info_imatge[1];
$tipus_mime = $info_imatge['mime']; // El més important

// 3. Aplicar validacions addicionals
if ($tipus_mime !== 'image/png' && $tipus_mime !== 'image/jpeg') {
    die("Error: Només s'accepten imatges PNG o JPEG. S'ha rebut: " . $tipus_mime);
}

if ($amplada > 1024) {
    die("Error: La imatge és massa ampla ($amplada px). Màxim permès: 1024px.");
}

// Si es supera la validació, es pot moure el fitxer...
// move_uploaded_file(...)
?>
```

`getimagesize()` vs. `mime_content_type()`

Ambdues funcions llegeixen el contingut del fitxer per determinar-ne el tipus, però tenen usos lleugerament diferents:

Característica	`mime_content_type()`	`getimagesize()`
Tipus de Fitxer	Qualsevol tipus de fitxer (PDF, JSON, imatge, etc.).	Només fitxers d'imatge (JPG, PNG, GIF, WebP, etc.).
Informació Principal	Només el tipus MIME (`string`).	Dimensions, MIME Type, i altres metadades de la imatge (`array`).
Validació de Seguretat	Bona: Identifica el tipus real.	**Excel·lent:** Identifica el tipus real **i** verifica la integritat de la capçalera de la imatge.

> **Recomanació:** Si espereu que l'usuari pugi **només imatges**, `getimagesize()` és la millor opció, ja que ofereix un control de seguretat més estricte sobre la validesa del format, a més de donar-vos les dimensions. Si accepteu una varietat de fitxers (imatges, PDFs, ZIPs), utilitzeu **`mime_content_type()`**.

Què és `finfo_open()`?

La funció **`finfo_open()`** és l'eina bàsica de l'extensió PHP **Fileinfo** que s'utilitza per obtenir informació detallada sobre el tipus de fitxer (MIME Type) i altres metadades, d'una manera més avançada i sovint més eficient que `mime_content_type()`, especialment si has de processar múltiples fitxers.

A diferència de `mime_content_type()`, que és una funció "d'un sol ús", **`finfo_open()`** inicialitza i retorna un **recurs d'identificació de fitxers** (un objecte `finfo`). Aquest recurs es pot reutilitzar per analitzar diversos fitxers sense haver de carregar les dades de les signatures MIME cada vegada.

La sintaxi bàsica és:

PHP

```
finfo_open(?int $flags = FILEINFO_NONE, ?string $magic_database = null): resource|
false
```

1. **`$flags` (Banderes):** Defineix quin tipus d'informació vols extreure.
   - **`FILEINFO_MIME` (Recomanat per a MIME):** Retorna la cadena del MIME Type (p. ex., `image/jpeg`). Aquesta és la bandera més utilitzada.
   - **`FILEINFO_MIME_TYPE`:** Retorna només el tipus MIME (sense el charset).
   - **`FILEINFO_MIME_ENCODING`:** Retorna l'encodificació MIME (p. ex., `binary`).
   - **`FILEINFO_NONE`:** Retorna una descripció llegible del contingut (p. ex., "JPEG image data, JFIF standard 1.01").
2. **`$magic_database`:** Opcionalment, pots especificar la ruta a un fitxer de base de dades magic personalitzat. Normalment es deixa a `null`, utilitzant el fitxer de base de dades del sistema per defecte.

Ús i exemples

L'ús de `finfo_open()` es divideix en tres passos: **Obrir, Analitzar i Tancar**.

Exemple: Obtenir MIME Type real d'un arxiu temporal

Aquest exemple és l'equivalent segur de `mime_content_type()`, utilitzat per validar fitxers pujats.

PHP

```php
<?php
// Assumim que hem rebut un fitxer temporal des de $_FILES
$fitxer_temporal = '/tmp/phpabc123'; // Substituir per $_FILES['arxiu']['tmp_name']

// 1. OBRIR: Inicialitzem el recurs per obtenir el MIME Type
// Utilitzem FILEINFO_MIME_TYPE per obtenir només la cadena 'tipus/subtipus' sense
charset
$finfo = finfo_open(FILEINFO_MIME_TYPE);

if ($finfo === false) {
    die("Error: No s'ha pogut inicialitzar la base de dades Fileinfo.");
}

// 2. ANALITZAR: Utilitzem finfo_file per examinar el fitxer temporal
$tipus_mime_real = finfo_file($finfo, $fitxer_temporal);

// 3. TANCAR: Alliberem el recurs un cop hem acabat. (Molt important per l'eficiència)
finfo_close($finfo);


// Validació:
if ($tipus_mime_real === 'image/jpeg' || $tipus_mime_real === 'image/png') {
    echo "Fitxer vàlid rebut. MIME Type real: " . $tipus_mime_real;
} else {
    echo "Error de seguretat. El tipus de fitxer '$tipus_mime_real' no està permès.";
}
?>
```

Exemple: Analitzar múltiples arxius (eficiència)

El principal avantatge de `finfo_open()` és que la inicialització (el pas 1) només es fa una vegada, fent que els passos posteriors d'anàlisi siguin més ràpids.

PHP

```php
<?php
// Llista de fitxers temporals (simulació de pujada múltiple)
$fitxers = [
    '/tmp/php1', // Suposem que és un PDF
    '/tmp/php2', // Suposem que és un ZIP
    '/tmp/php3'  // Suposem que és un JPEG
];

// 1. OBRIR una única vegada per obtenir el MIME Type
$finfo = finfo_open(FILEINFO_MIME_TYPE);

if ($finfo !== false) {
    echo "<h2>Resultats de l'Anàlisi</h2>";

    // 2. ANALITZAR tots els fitxers reutilitzant el recurs $finfo
```

```
    foreach ($fitxers as $path) {
        $tipus = finfo_file($finfo, $path);
        echo "<p>Fitxer $path: **$tipus**</p>";
    }

    // 3. TANCAR
    finfo_close($finfo);
}
?>
```

Avantatges

Funció	Avantatges	Desavantatges
`mime_content_type()`	Més simple i ràpida per a una sola comprovació. Codi concís.	La base de dades de signatures MIME es carrega cada vegada que es crida.
`finfo_open()`	**Més eficient** per processar múltiples fitxers. Ofereix més control sobre el tipus d'informació retornada (banderes).	Requereix tres passos (obrir, analitzar, tancar), fent el codi lleugerament més complex.

Recomanació: Com desenvolupador web, utilitzar `finfo_open()` és més **eficient** i és clau (processament de molts fitxers o en bucles), i `mime_content_type()` emprar-lo quan només necessites la validació ràpida d'un sol fitxer. Ambdues asseguren que la validació del MIME Type es faci mitjançant el **contingut real** del fitxer, i no mitjançant la informació no fiable del navegador.

Revisió núm. 5
★Admin l'ha creat 2025-10-10 10:36:24 UTC
Admin l'ha actualitzat 2025-10-14 10:20:24 UTC