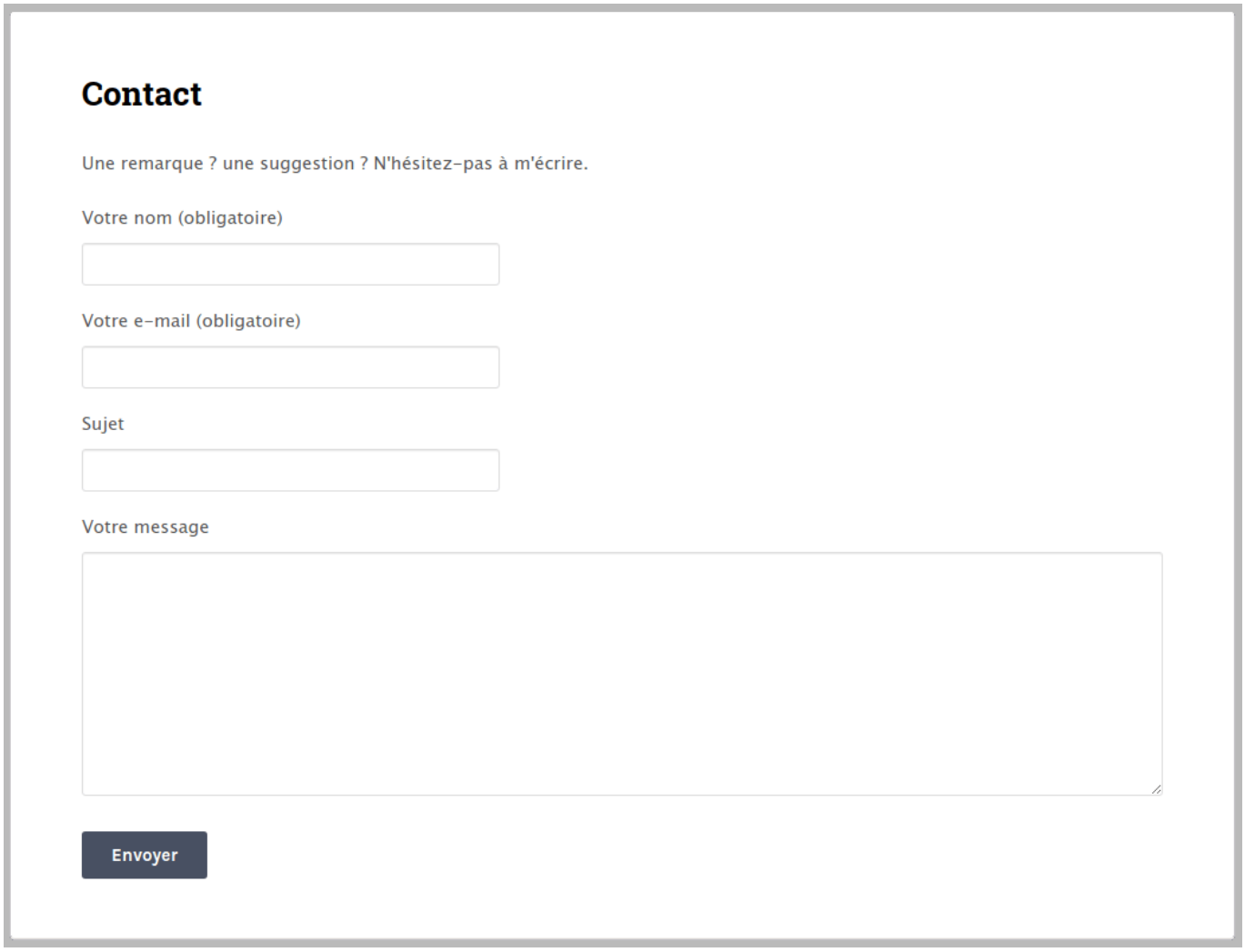
Envoi du contenu d'un formulaire vers un email

Il est difficile d'imaginer un site Web sans formulaire de contact. La difficulté ne réside pas vraiment dans le codage du formulaire en HTML mais plutôt dans la façon d'en récupérer les données puis de les recevoir par email.



La fonction PHP *mail()*

Une solution efficace est d'expédier le contenu du formulaire de contact vers un email destinataire à l'aide de la fonction PHP *mail()*. L'email du destinataire sera invisible, ce qui est un atout contre la collecte et le spam.

Voici un exemple simple utilisant un fichier au format HTML et un autre ficher au format PHP :

contact.html

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="utf-8">
    <title>Contactez-nous/title>
</head>

<body>
    <form method="post" action="mail.php">
        <label>Tape ton message ici pour m'écrire</label>
        <textarea rows="10" cols="30" name="message"></textarea>
        <input type="submit">
    </form>
```

```
</body>
</html>
```

La balise *<textarea>* permet de saisir plusieurs lignes de texte, là où la traditionnelle balise *<input type='text'>* n'autorise qu'une seule ligne de saisie. Ici, la méthode POST est préférable à la méthode GET afin d'éviter que les messages soient visibles dans l'historique du navigateur.

mail.php

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="utf-8">
    <title>Envoi d'un message par formulaire</title>
</head>

<body>
    <?php
    $retour = mail('destinataire@free.fr', 'Envoi depuis la page Contact', $_POST['message'],
'From: webmaster@monsite.fr');
    if ($retour)
        echo '<p>Votre message a bien été envoyé.</p>';
    ?>
</body>
</html>
```

Le contenu du formulaire est expédié vers l'email *destinataire@free.fr* grâce à la fonction *mail()*.

Syntaxe de la fonction *mail()* :

```
mail("Email du destinataire", "Sujet", "Message", "Entête");
```

Dans la chaîne *Entête*, le mot *From:* est suivi de l'email de l'expéditeur. Notez qu'on peut inventer l'email de l'expéditeur. Cet email peut être réel, ou pas. Cependant, pour éviter que votre message atterrisse dans le dossier *spam* du destinataire, il est conseillé de choisir un email d'expéditeur correspondant à votre nom de domaine, par exemple : *webmaster@monsite.fr*. Autre astuce, mettez une chaîne vide dans l'entête et le langage PHP choisira un email d'expéditeur correspondant à votre hébergeur :

```
$retour = mail('destinataire@free.fr', 'Envoi depuis la page Contact', $_POST['message'], '');
```

Page contact « tout en un »

Le code ci-dessous fusionne les pages précédentes grâce à la fonction *isset()* permettant de vérifier que le formulaire a été soumis. L'email de l'utilisateur saisi dans le champ *« Votre email »* devient l'adresse de réponse. Autre amélioration, les attributs *required* affectés aux champs de texte forceront l'utilisateur à renseigner ces champs.

contact.php

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="utf-8">
    <title>Contact</title>
</head>

<body>
    <h1>Contact</h1>
    <form method="post">
        <label>Votre email</label>
        <input type="email" name="email" required><br>
        <label>Message</label>
        <textarea name="message" required></textarea><br>
        <input type="submit">
```

```
    </form>
    <?php
    if (isset($_POST['message'])) {
        $retour = mail('destinataire@free.fr', 'Envoi depuis la page Contact',
$_POST['message'], 'From: webmaster@monsite.fr' . "\r\n" . 'Reply-to: ' . $_POST['email']);
        if($retour)
            echo '<p>Votre message a bien été envoyé.</p>';
    }
    ?>
</body>
</html>
```

Envoi d'un email au format HTML

Pour envoyer un email au format HTML, il faut mettre l'entête au format *MIME (Multipurpose Internet Mail Extensions).* Attention : les guillemets doubles autour des **« \r\n »** sont indispensables. Il permettent de générer des retours à la ligne.

L'intérêt est de pouvoir mettre des balises HTML dans le message. Des mises en forme seront alors possibles comme des caractères gras, des couleurs, des tableaux, des liens hypertextes, etc. Il sera également permis d'ajouter du code CSS.

```
<!DOCTYPE html>
<html lang="fr">

<head>
    <meta charset="utf-8">
    <title>Contact</title>
</head>

<body>
    <h1>Contactez-nous</h1>
    <form method="post">
        <label>Votre email</label>
        <input type="email" name="email" required>
        <label>Message</label>
        <textarea name="message" required></textarea>
        <input type="submit">
    </form>
    <?php
    if (isset($_POST['message'])) {
        $entete  = 'MIME-Version: 1.0' . "\r\n";
        $entete .= 'Content-type: text/html; charset=utf-8' . "\r\n";
        $entete .= 'From: webmaster@monsite.fr' . "\r\n";
        $entete .= 'Reply-to: ' . $_POST['email'];

        $message = '<h1>Message envoyé depuis la page Contact de monsite.fr</h1>
        <p><b>Email : </b>' . $_POST['email'] . '<br>
        <b>Message : </b>' . htmlspecialchars($_POST['message']) . '</p>';

        $retour = mail('destinataire@free.fr', 'Envoi depuis page Contact', $message,
$entete);
        if($retour)
            echo '<p>Votre message a bien été envoyé.</p>';
    }
    ?>
</body>
</html>
```

La fonction *htmlspecialchars()* permet de convertir les guillemets et autres apostrophes saisis dans le message en entités de caractère. Par exemple, un guillemet deviendra *"* et évitera de faire buguer le code PHP.

Pour éviter que votre message n'atterrisse dans les spams

L'email de l'expéditeur qui suit le préfixe *From:* doit être cohérent par rapport à votre hébergeur. Dans le cas contraire, les antispam du destinataire risquent de bloquer l'arrivée du message. Par exemple, si votre hébergeur est *OVH* et que vous mettez un expéditeur *gmail*, les antispam détecteront l'incohérence et votre message pourrait être bloqué.
Choisissez un email d'expéditeur en rapport avec votre domaine. Par exemple, si votre page Contact est hébergée sur le domaine *monsite.fr*, l'email de l'expéditeur pourrait être *webmaster@monsite.fr* ou *contact@monsite.fr*. Si vous souhaitez que la réponse du message se fasse sur l'email de l'utilisateur saisi dans le formulaire de la page contact, mettez l'email de réponse dans l'entête précédé de *Reply-to* comme montré dans le code ci-dessus.

Comme expliqué plus haut, vous pouvez aussi ne rien mettre à la ligne *From:* et le langage PHP choisira un email d'expédition correspondant à votre serveur :

```
$entete  = 'MIME-Version: 1.0' . "\r\n";
$entete .= 'Content-type: text/html; charset=utf-8' . "\r\n";
$entete .= 'Reply-to: ' . $_POST['email'];
```

La vidéo

Dans cette vidéo, je reprends la plupart des points présentés dans cet article : je décortique la fonction PHP *mail()* et je récupère les données d'un formulaire de contact avec la méthode *POST*.