Javascript

La gestion des erreurs en JavaScript

L'événement onerror

L'événement ***onerror*** est déclenché lorsqu'une erreur est rencontrée lors du chargement d'un fichier externe comme un document ou une image. Comme un exemple est toujours plus compréhensif que des explications textuelles, passons à un premier exemple. Nous allons tenter d'afficher une image sur notre page, une image qui n'existe pas dans les fichiers de notre site. Examinez le prochain bloc de code.

```
<body>
    <h1>La récupération des erreurs</h1>
    <img src="image1.gif"/>
    <span id="erreur"></span>
</body>
```

Puisque notre image est irrécupérable dans les fichiers de notre site, cette page affichera une petite icône à l'endroit où cette dernière est sensé apparaitre, comme le montre le résulatat de la prochaine figure.

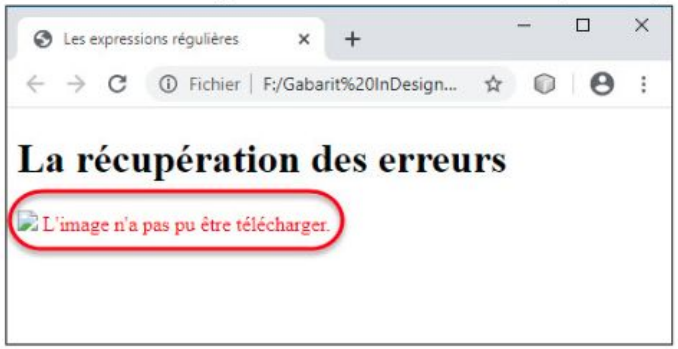
Nous allons maintenant modifier notre code afin de d'exécuter une méthode JavaScript que nous associerons à l'événement ***onerror*** de notre objet ***img***. Examinez la première modification apportée dans notre code.

```
<body>
    <h1>La récupération des erreurs</h1>
    <img src="image1.gif" onerror="maFonction()"/>
    <span id="erreur"></span>
</body>
```

Dans ce bloc de code, nous avons associé une fonction nommée ***maFonction()*** à l'événement ***onerror*** de notre élément ***img***. Nous allons maintenant créer notre fonction dans notre fichier script.js. Examinez le prochain bloc de code.

```
function maFonction()
{
    let erreur = document.getElementById("erreur");
    erreur.innerHTML = "L'image n'a pas pu être télécharger.";
    erreur.style.color = "red";
}
```

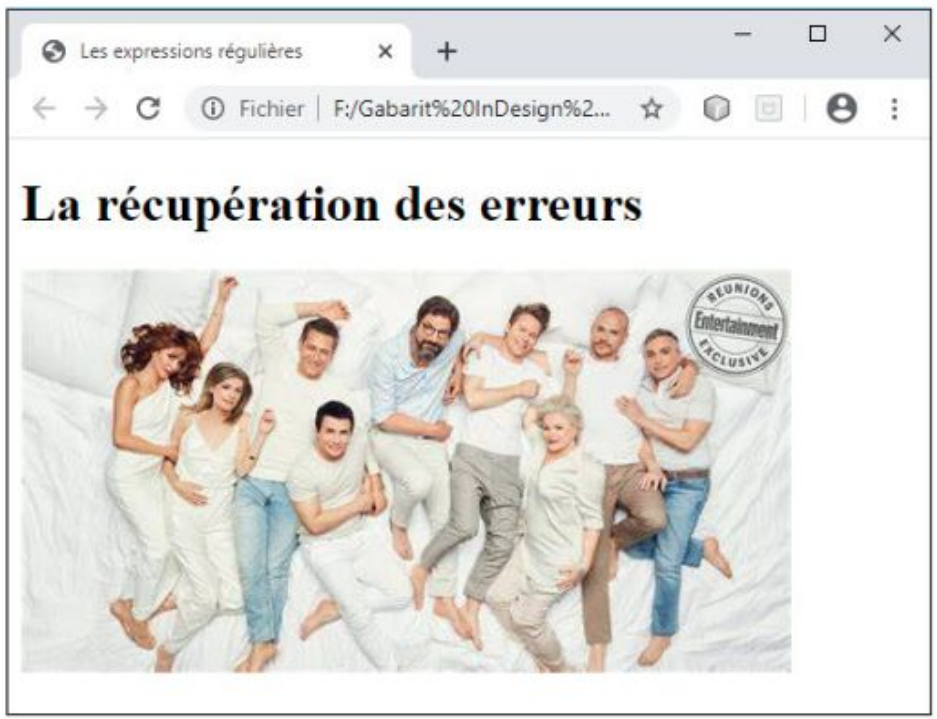
Cette fonction toute simple commence par récupérer notre élément ***span*** dont le ***id*** est ***erreur*** puis, nous ajoutons notre message et nous l'affichons en couleur rouge. Le résultat affiché à l'écran est celui de la prochaine figure.



JAVASCRIPT

Si nous modifions le lien de notre image afin de télécharger une image présente dans les fichiers de notre site, le message affiché précédemment ne sera pas affiché sur notre page, comme le montre le résultat de la prochaine figure.

```
<body>
    <h1>La récupération des erreurs</h1>
    <img src="Images/th.png" onerror="maFonction()"/>
    <span id="erreur"></span>
</body>
```



JAVASCRIPT

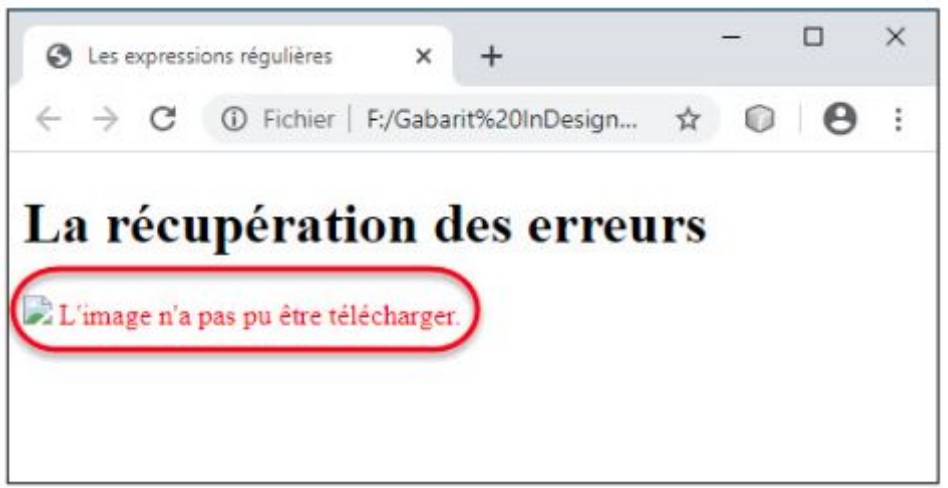
Tout comme nous l'avons vu précédemment, il est aussi possible d'utiliser la méthode ***addEventListener()*** de l'objet document afin d'associer notre fonction à notre événement ***onerror***. Examinez le prochain bloc de code qui utilise cette méthode et qui donnera les mêmes résultats affichés à l'écran que l'image soit téléchargée ou non.

```
<body>
    <h1>La récupération des erreurs</h1>
    <img id="image1" src="image1.gif"/>
    <span id="erreur"></span>
</body>

</html>

<script>
    document.getElementById("image1").addEventListener("error", maFonction);
</script>
```

Ce bloc de code ainsi modifié retournera le même résultat que celui que nous avions obtenu précédemment soit, celui de la prochaine figure.



Le bloc de code try ... catch

La syntaxe générale d'un bloc try ... catch est la suivante.

```
try
{
    // Bloc de code à exécuter.
}
catch(err)
{
    // Bloc de code à exécuter en cas d'erreur.
}
```

La première partie, la déclaration ***try***, vous permet de définir un bloc de code qui pourrait potentiellement contenir des erreurs lors de son exécution.

La deuxième partie, la déclaration ***catch***, vous permet de définir un bloc de code à exécuter si une erreur se produit dans le bloc ***try***.

Les déclarations ***try*** et ***catch*** en JavaScript viennent toujours en pair, c'est-à-dire que l'une ne va pas sans l'autre. Passons à un premier exemple afin de voir comment le tout s'exécute.

JAVASCRIPT

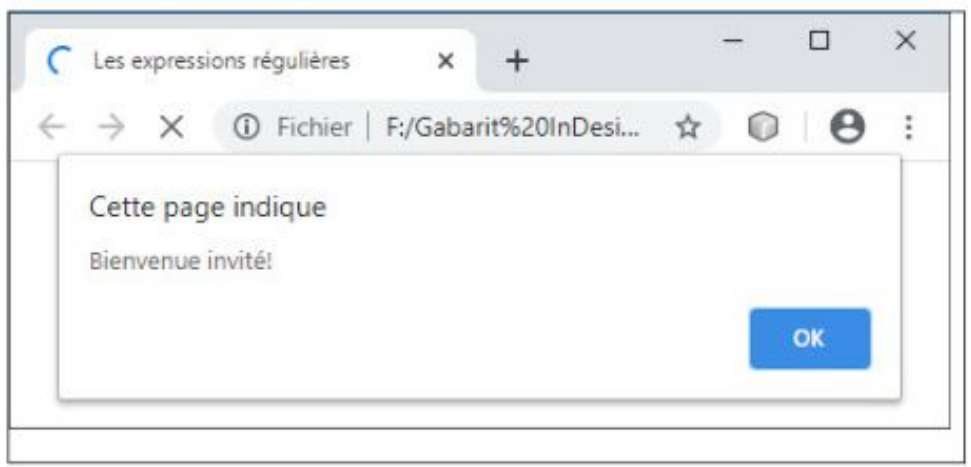
```
<body>
    <h1>La récupération des erreurs</h1>

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

    <script>
        alert("Bienvenue invité!");
    </script>

</body>
```

Ce petit bloc de code affiche tout simplement une boite de dialogue souhaitant la bienvenue. Le résultat qui sera affiché à l'écran sera celui de la prochaine figure.



JAVASCRIPT

Nous allons maintenant modifier notre code afin de récupérer une erreur de code que nous allons délibérément causer dans notre script. Examinez le prochain bloc de code.

```
<body>
    <h1>La récupération des erreurs</h1>

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

    <script>
        try
        {
            adlert("Bienvenue invité!");
        }
        catch(err)
        {
            let erreur = document.getElementById("demo")
            erreur.innerHTML = err.name + "<br/>" + err.message;
            erreur.style.color = "red";
        }
    </script>
</body>
```

Dans ce bloc de code, nous avons modifié la méthode ***alert()*** et avons écrit ***adlert()***. En JavaScript, lorsqu'une erreur est rencontrée, le script s'arrête et un objet de type ***error*** est généré. Cet objet contient deux informations que l'on peut récupérer soit, le nom de l'erreur et un message. Cet objet est récupéré à l'aide d'une variable passée en paramètre à notre déclaration ***catch***, dans cet exemple, ***err***. L'exécution de ce bloc de code affichera le résultat de la prochaine figure à l'écran.

JAVASCRIPT

Il est possible de modifier les valeurs par défaut retournées par les propriétés ***name*** et ***message*** de l'objet ***error***. Examinez le prochain bloc de code.

```
<script>
    try
    {
        adlert("Bienvenue invité!");
    }
    catch(err)
    {
        err.name = "Erreur de référence :";
        err.message = "L'instruction adlert() n'est pas une instruction valide."
        let erreur = document.getElementById("demo")
        erreur.innerHTML = err.name + "<br/>" + err.message;
        erreur.style.color = "red";
    }
</script>
```