jQuery/JavaScript to replace broken images

Asked 10 years, 10 months ago Active 24 days ago Viewed 240k times

516

★
228

I have a web page that includes a bunch of images. Sometimes the image isn't available, so a broken image is displayed in the client's browser.

How do I use jQuery to get the set of images, filter it to broken images then replace the src?

--I thought it would be easier to do this with jQuery, but it turned out much easier to just use a pure JavaScript solution, that is, the one provided by Prestaul.

javascript jquery html brokenimage

edited May 15 '16 at 19:54
Peter Mortensen
14.3k 19 88 116

asked Sep 18 '08 at 13:53
Dan Lord
2,846 3 14 15

30 Answers

718

✓

Handle the `onError` event for the image to reassign its source using JavaScript:

```
function imgError(image) {
    image.onerror = "";
    image.src = "/images/noimage.gif";
    return true;
}
```

```
<img src="image.png" onerror="imgError(this);"/>
```

Or without a JavaScript function:

```
<img src="image.png" onError="this.onerror=null;this.src='/images/noimage.gif';" />
```

The following compatibility table lists the browsers that support the error facility:

http://www.quirksmode.org/dom/events/error.html

edited Apr 28 '13 at 14:58
Ry- ♦
175k 42 358 376

answered Sep 18 '08 at 14:04
Prestaul
68.2k 9 71 81

127 You probably want to clear out onerror *before* setting src. Otherwise if noimage.gif is also missing you might end up with a "stack overflow". – pcorcoran Sep 18 '08 at 17:14

43 I thought & hoped we were moving away from inline attributes for javascript events... – redsquare Sep 2 '09 at 6:04

23 redsquare... they are perfectly understandable when you are judicious about keeping them under control and can even be useful when you want your markup to reflect what is actually going to happen. – Nicole Feb 18 '10 at 21:41

2 Good point NickC, about the markup showing what's happening, I just cringe when I look at it :P – SSH This Mar 27 '12 at 21:11

35 @redsquare, I completely agree, but this is a unique case. Inline is the only place where you can attach the listener with absolute certainty that the event will not trigger before your listener is attached. If you do it at the end of the document or wait for DOM loaded you may miss the error event on some images. It does you no good to attach an error handler after the event has fired. – Prestaul Sep 5 '12 at 0:40

191

I use the built in `error` handler:

```
$("img").error(function () {
    $(this).unbind("error").attr("src", "broken.gif");
});
```

Edit: The `error()` method is deprecated in jquery 1.8 and higher. Instead, you should use `.on("error")` instead:

```
$("img").on("error", function () {
    $(this).attr("src", "broken.gif");
});
```

edited Jun 28 at 12:06
J M

answered Oct 3 '08 at 19:18
travis

232 3 14 **26.5k** 20 63 92

107 If you use this technique you can use the "one" method to avoid needing to unbind the event: $('img').one('error', function() { this.src = 'broken.gif'; }); – Prestaul Apr 1 '11 at 16:18

7 +1 But do you have to unbind it? It works beautifully on dynamic images (eg. ones that can change on hover) if you don't unbind it. – Aximili Apr 5 '11 at 5:27

15 I think that if you don't unbind it, and the broken.gif src reference ever fails to load for any reason, then bad things could happen in the browser. – travis Apr 5 '11 at 15:49

4 @travis, at what point would you make this call? Is there any way to use this method and be certain that your handler is getting attached BEFORE the error event fires? – Prestaul Sep 5 '12 at 0:43

5 Here is an example of a case where the error fires before the event handler is attached: jsfiddle.net/zmpGz/1 I'm genuinely curious if there is a fail-safe way to attach this handler... – Prestaul Sep 5 '12 at 1:03

114

In case someone like me, tries to attach the `error` event to a dynamic HTML `img` tag, I'd like to point out that, there is a catch:

Apparently `img` error events **don't bubble** in most browsers, contrary to what the standard says.

So, something like the following will **not work**:

```
$(document).on('error', 'img', function () { ... })
```

Hope this will be helpful to someone else. I wish I had seen this here in this thread. But, I didn't. So, I am adding it

edited Jun 14 '16 at 13:19
Mohd Abdul Mujib
7,183 4 41 67

answered Jun 15 '12 at 13:54
mouad
48.8k 14 95 95

didn't work. I load a dead image dynamically and append it to the dom, and no 'error' event is fired. – vsync Jun 14 '14 at 19:51

6 damn - i didn´t notice that is is a example how not to do it .. – Benjamin Aug 11 '15 at 17:09

Here is a standalone solution:

55

```
$(window).load(function() {
  $('img').each(function() {
    if (!this.complete || typeof this.naturalWidth == "undefined" || this.naturalWidth
== 0) {
      // image was broken, replace with your new image
      this.src = 'http://www.tranism.com/weblog/images/broken_ipod.gif';
    }
  });
});
```

edited Sep 18 '08 at 14:38

answered Sep 18 '08 at 14:26

Devon
4,529 4 34 44

5 Almost correct... a correct image in IE will still return (typeof(this.naturalWidth) == "undefined") == true; - I changed the if statement to (!this.complete || (!$.browser.msie && (typeof this.naturalWidth == "undefined" || this.naturalWidth == 0))) – Sugendran Feb 19 '09 at 2:42

3 this rules because it can detect a broken image after a page load. i needed this. – northamerican Sep 30 '13 at 14:51

@Sugendran `$.browser` has been deprecated and removed now, use feature detection instead (gets more complicated in this case). – lee penkman Dec 3 '14 at 22:20

3 Standalone but requires jQuery? – Charlie Jul 6 '16 at 18:12

This works great, but when the src of the image returns a `204 No Content` it would be a broken image. – Carson Reinke Oct 4 '16 at 14:59

32

I believe this is what you're after: jQuery.Preload

Here's the example code from the demo, you specify the loading and not found images and you're all set:

```
$('#images img').preload({
    placeholder:'placeholder.jpg',
    notFound:'notfound.jpg'
});
```

answered Sep 18 '08 at 14:05

Nick Craver ♦
548k 116 1218
1118

1 jQuery.Preload is a plugin which is required to use the example code. – Dan Lord Sep 18 '08 at 15:35

17 Yes...this is linked in the answer on the first line. – Nick Craver ♦ Sep 18 '08 at 16:03

Thank you Nick, this appears to be a good solution for the 8+ year old problem of Firefox not showing an image replacement for broken images. Plus you can brand the little thing...thanks again. – klewis Aug 8 '14 at 12:24

23

```
$(window).bind('load', function() {
$('img').each(function() {
    if((typeof this.naturalWidth != "undefined" &&
        this.naturalWidth == 0 )
        || this.readyState == 'uninitialized' ) {
        $(this).attr('src', 'missing.jpg');
    }
}); })
```

Source: http://www.developria.com/2009/03/jquery-quickie---broken-images.html

edited Sep 16 '13 at 19:49
XP84
734 7 15

answered Mar 25 '10 at 23:57
Mohamad
239 2 2

1 Fixed issue of onerror not firing on firefox :) Thanks. – ram Jun 25 '14 at 13:13

12

While the OP was looking to replace the SRC, I'm sure many people hitting this question may only wish to hide the broken image, in which case this simple solution worked great for me:

```
<img src="img.jpg" onerror="this.style.display='none';" />
```

Run code snippet   Expand snippet

edited Jul 12 at 14:34

answered Oct 26 '16 at 13:41
Nathan Arthur

3,316 3 26 52

1 I have found this as a most reliable solution for me to a very similar problem. I just rather to not show any image than show a broken image alternative. Thanks. – pc_ Feb 4 '18 at 23:07

10

Here is a quick-and-dirty way to replace all the broken images, and there is no need to change the HTML code ;)

codepen example

```
  $("img").each(function(){
      var img = $(this);
      var image = new Image();
      image.src = $(img).attr("src");
      var no_image = "https://dummyimage.com/100x100/7080b5/000000&text=No+image";
      if (image.naturalWidth == 0 || image.readyState == 'uninitialized'){
          $(img).unbind("error").attr("src", no_image).css({
              height: $(img).css("height"),
              width: $(img).css("width"),
          });
      }
});
```

edited Sep 9 '18 at 23:13

answered Feb 18 '14 at 15:59

Ashfaq Ahmed

1,193 3 16 30

8

This is a crappy technique, but it's pretty much guaranteed:

```
<img ...  onerror="this.parentNode.removeChild(this);">
```

answered Apr 11 '14 at 13:54

Phil LaNasa

2,413 20 15

8

I couldn't find a script to suit my needs, so I made a recursive function to check for broken images and attempt to reload them every four seconds until they are fixed.

I limited it to 10 attempts as if it's not loaded by then the image might not be present on server and the function would enter an infinite loop. I am still testing though. Feel free to tweak it :)

```
var retries = 0;
$.imgReload = function() {
    var loaded = 1;

    $("img").each(function() {
        if (!this.complete || typeof this.naturalWidth == "undefined" ||
this.naturalWidth == 0) {

            var src = $(this).attr("src");
            var date = new Date();
            $(this).attr("src", src + "?v=" + date.getTime()); //slightly change url to
prevent loading from cache
            loaded =0;
        }
    });

    retries +=1;
    if (retries < 10) { // If after 10 retries error images are not fixed maybe because
they
                        // are not present on server, the recursion will break the loop
        if (loaded == 0) {
            setTimeout('$.imgReload()',4000); // I think 4 seconds is enough to load a
small image (<50k) from a slow server
        }
        // All images have been loaded
        else {
            // alert("images loaded");
        }
    }
    // If error images cannot be loaded  after 10 retries
    else {
        // alert("recursion exceeded");
    }
}

jQuery(document).ready(function() {
    setTimeout('$.imgReload()',5000);
});
```

edited May 15 '16 at 19:59

answered Dec 20 '11 at 12:15

Peter Mortensen
14.3k 19 88 116

Shade
81 1 1

Nice idea with trying to reload images. This is useful for images that are uploaded/autogenerated and the server might not have gotten/completed as the page is loaded. That's some original formatting. A bit too inconsistent and not in my taste however... – Joakim Nov 10 '14 at 13:04

5

You can use GitHub's own fetch for this:

Frontend: https://github.com/github/fetch
or for Backend, a Node.js version: https://github.com/bitinn/node-fetch

```
fetch(url)
  .then(function(res) {
    if (res.status == '200') {
      return image;
    } else {
      return placeholder;
    }
  }
```

Edit: This method is going to replace XHR and supposedly already has been in Chrome. To anyone reading this in the future, you may not need the aforementioned library included.

edited Sep 22 '16 at 18:18

answered May 2 '16 at 19:54
BarryMode
758 1 9 24

5

This is JavaScript, should be cross browser compatible, and delivers without the ugly markup `onerror=""` :

```
var sPathToDefaultImg = 'http://cdn.sstatic.net/stackexchange/img/logos/so/so-icon.png',
    validateImage = function( domImg ) {
        oImg = new Image();
        oImg.onerror = function() {
            domImg.src = sPathToDefaultImg;
        };
        oImg.src = domImg.src;
    },
    aImg = document.getElementsByTagName( 'IMG' ),
    i = aImg.length;

while ( i-- ) {
    validateImage( aImg[i] );
}
```

CODEPEN:

edited Sep 5 '17 at 10:31

answered Aug 19 '14 at 6:53
Axel
2,286 10 23 41

4

Better call using

```
jQuery(window).load(function(){
    $.imgReload();
});
```

Because using `document.ready` doesn't necessary imply that images are loaded, only the HTML. Thus, there is no need for a delayed call.

edited May 15 '16 at 20:00
Peter Mortensen
14.3k 19 88 116

answered Dec 23 '11 at 23:04
Shade
41 1

4

CoffeeScript variant:

I made it to fix an issue with Turbolinks that causes the .error() method to get raised in Firefox sometimes even though the image is really there.

```
$("img").error ->
  e = $(@).get 0
  $(@).hide() if !$.browser.msie && (typeof this.naturalWidth == "undefined" ||
this.naturalWidth == 0)
```

edited May 15 '16 at 20:01
Peter Mortensen
14.3k 19 88 116

answered Jul 18 '13 at 17:24
Kevin
3,309 1 28 37

4

By using Prestaul's answer, I added some checks and I prefer to use the jQuery way.

```
<img src="image1.png" onerror="imgError(this,1);"/>
<img src="image2.png" onerror="imgError(this,2);"/>

function imgError(image, type) {
    if (typeof jQuery !== 'undefined') {
       var imgWidth=$(image).attr("width");
       var imgHeight=$(image).attr("height");

        // Type 1 puts a placeholder image
        // Type 2 hides img tag
        if (type == 1) {
            if (typeof imgWidth !== 'undefined' && typeof imgHeight !== 'undefined') {
                $(image).attr("src", "http://lorempixel.com/" + imgWidth + "/" +
imgHeight + "/");
            } else {
               $(image).attr("src", "http://lorempixel.com/200/200/");
            }
        } else if (type == 2) {
            $(image).hide();
        }
    }
    return true;
}
```

edited May 23 '17 at 11:55
Community ♦
1 1

answered Jun 29 '14 at 10:28
trante
22.8k 40 158 250

4

If you have inserted your `img` with `innerHTML`, like: `$("div").innerHTML = <img src="wrong-uri">`, you can load another image if it fails doing, e.g, this:

```
<script>
    function imgError(img) {
        img.error="";
        img.src="valid-uri";
    }
</script>

<img src="wrong-uri" onerror="javascript:imgError(this)">
```

Why is `javascript: _` needed? Because scripts injected into the DOM via script tags in `innerHTML` are not run at the time they are injected, so you have to be explicit.

edited May 15 '16 at 20:10
Peter Mortensen
14.3k 19 88 116

answered Nov 12 '14 at 12:22
horro
485 2 8 28

4

I found this post while looking at this other SO post. Below is a copy of the answer I gave there.

I know this is an old thread, but React has become popular and, perhaps, someone using React will come here looking for an answer to the same problem.

So, if you are using React, you can do something like the below, which was an answer original provided by Ben Alpert of the React team here

```
getInitialState: function(event) {
    return {image: "http://example.com/primary_image.jpg"};
},
handleError: function(event) {
    this.setState({image: "http://example.com/failover_image.jpg"});
},
render: function() {
    return (
        <img onError={this.handleError} src={src} />;
    );
}
```

edited May 23 '17 at 11:55
Community ♦
1 1

answered Jul 13 '16 at 18:28
Jordan Bonitatis
1,194 10 10

4

I created a fiddle to replace the broken image using "onerror" event. This may help you.

```
    //the placeholder image url
    var defaultUrl = "url('https://sadasd/image02.png')";

    $('div').each(function(index, item) {
      var currentUrl = $(item).css("background-image").replace(/^url\(['"](.+)['"]\)/,
'$1');
      $('<img>', {
        src: currentUrl
      }).on("error", function(e) {
        $this = $(this);
        $this.css({
          "background-image": defaultUrl
        })
        e.target.remove()
      }.bind(this))
    })
```

answered Sep 1 '16 at 13:23

Ninjaneer

1,493 10 20

4

Here is an example using the HTML5 Image object wrapped by JQuery. Call the load function for the primary image URL and if that load causes an error, replace the src attribute of the image with a backup URL.

```
function loadImageUseBackupUrlOnError(imgId, primaryUrl, backupUrl) {
    var $img = $('#' + imgId);
    $(new Image()).load().error(function() {
        $img.attr('src', backupUrl);
    }).attr('src', primaryUrl)
}

<img id="myImage" src="primary-image-url"/>
<script>
    loadImageUseBackupUrlOnError('myImage','primary-image-url','backup-image-url');
</script>
```

answered Nov 4 '16 at 16:00

Kurt Hartmann

74 5

4

Pure JS. My task was: if image 'bl-once.png' is empty -> insert the first one (that hasn't 404 status) image from array list (in current dir):

```
<img src="http://localhost:63342/GetImage/bl-once.png" width="200"
onerror="replaceEmptyImage.insertImg(this)">
```

Run code snippet Expand snippet

Maybe it needs to be improved, but:

```
var srcToInsertArr = ['empty1.png', 'empty2.png', 'needed.png', 'notActual.png']; // try
to insert one by one img from this array
    var path;
    var imgNotFounded = true; // to mark when success

    var replaceEmptyImage = {
        insertImg: function (elem) {

            if (srcToInsertArr.length == 0) { // if there are no more src to try return
                return "no-image.png";
            }
            if(!/undefined/.test(elem.src)) { // remember path
                path = elem.src.split("/").slice(0, -1).join("/"); //
"http://localhost:63342/GetImage"
            }
            var url = path + "/" + srcToInsertArr[0];

            srcToInsertArr.splice(0, 1); // tried 1 src


                if(imgNotFounded){ // while not success
                    replaceEmptyImage.getImg(url, path, elem); // CALL GET IMAGE
                }


        },
        getImg: function (src, path, elem) { // GET IMAGE

            if (src && path && elem) { // src =
"http://localhost:63342/GetImage/needed.png"
```

```
                var pathArr = src.split("/"); // ["http:", "", "localhost:63342",
"GetImage", "needed.png"]
                var name = pathArr[pathArr.length - 1]; // "needed.png"

                xhr = new XMLHttpRequest();
                xhr.open('GET', src, true);
                xhr.send();

                xhr.onreadystatechange = function () {

                    if (xhr.status == 200) {
                        elem.src = src; // insert correct src
                        imgNotFounded = false; // mark success
                    }
                    else {
                        console.log(name + " doesn't exist!");
                        elem.onerror();
                    }

                }
            }
        }

    };
```

Run code snippet Expand snippet

So, it will insert correct 'needed.png' to my src or 'no-image.png' from current dir.

edited Apr 10 '17 at 13:59

answered Apr 7 '17 at 15:10
Дмитрий Дорогонов
520 1 5 10

Is it working cross-domain? – jasxir Jun 23 '17 at 9:38

I've used on the same site. This is just idea. My real code was improved... – Дмитрий Дорогонов Jun 23 '17 at 12:44

3

I am not sure if there is a better way, but I can think of a hack to get it - you could Ajax post to the img URL, and parse the response to see if the image actually came back. If it came back as a 404 or something, then swap out the img. Though I expect this to be quite slow.

edited May 15 '16 at 19:55
Peter Mortensen
14.3k 19 88 116

answered Sep 18 '08 at 14:05
Chii
12k 2 29 44

3

I solved my problem with these two simple functions:

```
function imgExists(imgPath) {
    var http = jQuery.ajax({
                   type:"HEAD",
                   url: imgPath,
                   async: false
               });
    return http.status != 404;
}

function handleImageError() {
    var imgPath;

    $('img').each(function() {
        imgPath = $(this).attr('src');
        if (!imgExists(imgPath)) {
            $(this).attr('src', 'images/noimage.jpg');
        }
    });
}
```

edited May 15 '16 at 20:03
Peter Mortensen
14.3k 19 88 116

answered Oct 21 '13 at 11:11
Luis Gustavo Beligante
41 1

3

jQuery 1.8

```
// If missing.png is missing, it is replaced by replacement.png
$( "img" )
  .error(function() {
    $( this ).attr( "src", "replacement.png" );
  })
  .attr( "src", "missing.png" );
```

jQuery 3

```
// If missing.png is missing, it is replaced by replacement.png
$( "img" )
  .on("error", function() {
    $( this ).attr( "src", "replacement.png" );
  })
  .attr( "src", "missing.png" );
```

reference

answered Sep 2 '17 at 3:16
Nabi K.A.Z.
3,427 1 30 43

2

```
;(window.jQuery || window.Zepto).fn.fallback = function (fallback) {
    return this.one('error', function () {
        var self = this;
        this.src = (fallback || 'http://lorempixel.com/$width/$height')
        .replace(/\$(\w+)/g, function (m, t) { return self[t] || ''; });
    });
};
```

You can pass a placeholder path and acces in it all properties from the failed image object via `$*` :

```
$('img').fallback('http://dummyimage.com/$widthx$height&text=$src');
```

http://jsfiddle.net/ARTsinn/Cu4Zn/

edited Jul 1 '13 at 11:02

answered Jun 27 '13 at 20:34
yckart
22k 5 99 113

This has been frustrating me for years. My CSS fix sets a background image on the `img` . When a dynamic image `src` doesn't load to the foreground, a placeholder is visible on the `img` 's bg. This works if your images have a default size (e.g. `height` , `min-height` , `width` and/or `min-width`).

2

You'll see the broken image icon but it's an improvement. Tested down to IE9 successfully. iOS Safari and Chrome don't even show a broken icon.

```
.dynamicContainer img {
  background: url('/images/placeholder.png');
  background-size: contain;
}
```

Add a little animation to give `src` time to load without a background flicker. Chrome fades in the background smoothly but desktop Safari doesn't.

```
.dynamicContainer img {
  background: url('/images/placeholder.png');
  background-size: contain;
  -webkit-animation: fadein 1s;
  animation: fadein 1s;
}

@-webkit-keyframes fadein {
  0%   { opacity: 0.0; }
  50%  { opacity: 0.5; }
  100% { opacity: 1.0; }
}

@keyframes fadein {
  0%   { opacity: 0.0; }
  50%  { opacity: 0.5; }
  100% { opacity: 1.0; }
}
```

answered Apr 25 '14 at 15:20

Dylan Valade

4,202 5 34 53

2

If the image cannot be loaded (for example, because it is not present at the supplied URL), image URL will be changed into default,

For more about [.error()](#)

```
$('img').on('error', function (e) {
  $(this).attr('src', 'broken.png');
});
```

Run code snippet Expand snippet

answered Feb 20 '18 at 4:41

CasperSL
426 3 8

2

I think I have a more elegant way with event delegation and event capturing on `window`'s `error` even when the backup image fail to load.

```
img {
  width: 100px;
  height: 100px;
}



<script>
  window.addEventListener('error', windowErrorCb, {
    capture: true
  }, true)

  function windowErrorCb(event) {
    let target = event.target
    let isImg = target.tagName.toLowerCase() === 'img'
    if (isImg) {
      imgErrorCb()
      return
    }

    function imgErrorCb() {
      let isImgErrorHandled = target.hasAttribute('data-src-error')
      if (!isImgErrorHandled) {
        target.setAttribute('data-src-error', 'handled')
        target.src = 'backup.png'
      } else {
        //anything you want to do
        console.log(target.alt, 'both origin and backup image fail to load!');
      }
    }
  }
</script>
<img id="img" src="error1.png" alt="error1">
<img id="img" src="error2.png" alt="error2">
<img id="img" src="https://i.stack.imgur.com/ZXCE2.jpg" alt="avatar">
```

Run code snippet   Expand snippet

The point is :

1. Put the code in the `head` and executed as the first inline script. So, it will listen the errors happen after the script.
2. Use event capturing to catch the errors, especially for those events which don't bubble.
3. Use event delegation which avoids binding events on each image.
4. Give the error `img` element an attribute after giving them a `backup.png` to avoid disappearance of the `backup.png` and subsequent infinite loop like below:

img error->backup.png->error->backup.png->error->,,,,,

edited May 24 '18 at 1:30

answered May 23 '18 at 9:37
xianshenglu
3,150 1 5 21

1

I got the same problem. This code works well on my case.

```
// Replace broken images by a default img
$('img').each(function(){
    if($(this).attr('src') === ''){
      this.src = '/default_feature_image.png';
    }
});
```

answered Nov 16 '17 at 20:43
Dale Nguyen
797 9 20

1

Sometimes using the `error` event is not feasible, e.g. because you're trying to do something on a page that’s already loaded, such as when you’re running code via the console, a bookmarklet, or a script loaded asynchronously. In that case, checking that `img.naturalWidth` and `img.naturalHeight` are 0 seems to do the trick.

For example, here's a snippet to reload all broken images from the console:

```
$$("img").forEach(img => {
  if (!img.naturalWidth && !img.naturalHeight) {
    img.src = img.src;
  }
}
```

answered Jan 14 '18 at 15:56

Lea Verou

18.4k 8 37 45

0

I found this to work best, if any image fails to load the first time, it is completely removed from the DOM. Executing `console.clear()` keeps the console window clean, since the 404 errors cannot be omitted with try/catch blocks.

```
$('img').one('error', function(err) {
    // console.log(JSON.stringify(err, null, 4))
    $(this).remove()
    console.clear()
})
```

edited Feb 1 at 15:55

answered Feb 1 at 0:38

indospace.io

374 5 14

protected by Samuel Liew ♦ Nov 28 '16 at 3:21

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site (the association bonus does not count).

Would you like to answer one of these unanswered questions instead?