Enlargeit! Version 1.1 Operation Manual

Contents **Page**

Enlargeit! Version 1.1 Operation Manual

EnlargeIt! v1.1 – © 2008 Timo Sack - http://enlargeit.timos-welt.de

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. See LICENSE.TXT for details.

1 What is EnlargeIt!

EnlargeIt! is a small Javascript file you can use to

- enlarge images beautifully in your web pages,
- easily embed SWF flash animations into your pages,
- present HTML content as iframe windows or to
- display a bunch of movie types (FLV, YouTube, DivX).

You may configure it to your needs in many ways; you can use different animations, border, shadow and much more. You can add AJAX content to your files, e. g. as description, and link between AJAX snippets.

EnlargeIt! is very light weight (max. 22 Kbyte) and easy to integrate. If you like, you can integrate it with full fall back option for those who can't use Javascript, either because they have turned it off in their browsers, or because it is filtered by a company firewall.

2 What does EnlargeIt! need

EnlargeIt! needs a thumbnail picture to display your file. Any web format will work (GIF, JPEG, PNG). It will not generate the thumbnail pics for you, you'll have to do this yourself. There are plenty of freeware solutions for this purpose on the web, just use Google.

If you use *EnlargeIt!* to display images, it will work without thumbnail images as long as you use the full size image with the attributes *width* and *height* as scaled down thumbnail, but this is not recommended because of the long loading times. *EnlargeIt!* has to wait to enlarge the first image until the *whole* page has ready loaded, that means including all images.

3 Displaying pictures

3.1 Easy integration

This will be the way to integrate EnlargeIt!, if you like to keep things very small and easy. If you chose easy integration, visitors without Javascript (either because they turned it off in their browsers, or because it is filtered out by a company firewall) won't be able to enlarge the pictures. They'll just see the thumbnail image and nothing will happen if they click on it.

If your site is heavily based on Javascript and won't work without it anyways, or if your site is of private nature and won't be visited by company people, then this is the way to go. Otherwise, please do a failsafe integration.

Most people will check the examples inside the *EnlargeIt!* ZIP file and understand how it must be integrated within seconds. For the rest of us (or for people who in general prefer to read the fine manual before trying out), here are the detailed integration steps:

Step 1: Paste this to the <*head*> section of your web page:

```
<script type="text/javascript" src="path/to/enlargeit.js"></script>
```

Replace *"path/to/"* with the actual path where enlargeit.js will be placed on your server.

Step 2: Integrate your thumbnails like this into the <body> of your web page:

```
<img src="images/thumb_file.jpg" alt="My first picture" onclick="enlarge(this);"
longdesc="images/fullsize_file.jpg" />
```

The attribute *longdesc* is used for the qualified filename of the full size image, meaning that if the file is in a different folder, the correct path must be included (in the example, it's *images/*). For other files than pictures (movies, flash files, iframes), the *longdesc* attribute must follow a different syntax; please have a look at the appropriate chapters of this documentation.

The attribute *alt* is used for the picture caption. If you don't define it or leave it empty, no caption (title bar text) will be displayed for this picture.

Step 3: Copy the files that belong to *EnlargeIt!* (you'll find them in the ZIP file inside the folder */files*) to a defined place on your web space. Normally you simply upload the whole folder.

Step 4: Open *enlargeit.js* and enter the correct path for value *enl_gifpath*, meaning that you enter the path to the place where the files from Step 3 are, seen from the place where the actual web page is. Standard value is '*./files/*', meaning that EnlargeIt! will search for the graphic files in a subfolder *files* of your web page. The last character of *enl_gifpath* must be a slash (/).

That's it, EnlargeIt! should work already, though not all settings may be as you wish. Please check chapter 9 ('Settings in detail') for more information.

3.2 Failsafe integration

If you like to ensure, that visitors without Javascript (either because they turned it off in their browsers, or because it is filtered out by a company firewall) will be able to see the full size versions of your images, you have to chose failsafe integration.

Most people will check the examples inside the EnlargeIt! ZIP file and understand how it must be integrated within seconds. For the rest of us (or people who in general prefer to read the fine manual before trying out), here are the detailed integration steps:

Step 1: Paste this to the <*head*> section of your web page:

```
<script type="text/javascript" src="path/to/enlargeit.js"></script>
```

Replace *"path/to/"* with the actual path where *enlargeit.js* will be placed on your server.

Step 2: Integrate your thumbnails like this into the <*body*> of your web page:

```
<a href=”images/fullsize_file.jpg” target=”_blank” onclick=”return false;”><img
src="images/thumb_file.jpg" alt="My first picture" onclick="enlarge(this);"
longdesc="images/fullsize_file.jpg" /></a>
```

The attribute *longdesc* is used for the qualified filename of the full size image, meaning that if the file is in a different folder, the correct path must be included (in the example, it's *images/*).

The attribute *alt* is used for the picture caption. If you don't define it or leave it empty, no caption (title bar text) will be displayed for this picture.

Around the image, there's an <*a*>…</*a*> element linking to the full size image in a new window (*target=”_blank”*). It's very important, that you set the attribute *onclick=”return false;”*. This way, the link won't be used if Javascript is available.

Step 3: Copy the files that belong to *EnlargeIt!* (you'll find them in the ZIP file inside the folder */files*) to a defined place on your web space. Normally you'll simply upload the whole folder.

Step 4: Open *enlargeit.js* and enter the correct path for value *enl_gifpath*, meaning that you enter the path to the place where the files from Step 3 are, seen from the place where the actual web page is. Standard value is '*./files/*', meaning that EnlargeIt! will search for the graphic files in a subfolder *files* of your web page. The last character of *enl_gifpath* must be a slash (/).

That's it, EnlargeIt! should work already, though not all settings may be as you wish. Please have a look at chapter 9 ('Settings in detail') for more information.

4 Displaying flash (.swf) files*

It's as easy as displaying an image (see chapter 3). Here are the differences:

- EnlargeIt! checks if the visitor has a flash plug in installed. If not, a message is displayed and nothing will be enlarged.
- The *longdesc* attribute must look like this:
  longdesc="swf::path/to/flashfile::width::height"
  Example:
  longdesc="swf::subfolder/myfile.swf::450::400"
- It is not possible to use an animation when displaying flash files. EnlargeIt! will never use glide or fade to display a flash file, no matter what you set *enl_ani* to.
- You shoudn't set *enl_darksteps* too high if you want to display flash files, because it can make the playback stutter on slow computers. Recommended is a value < 5.

Flash files behave exactly like images – they can have the same buttons, they can be grouped via class attribute, they support drag & drop if you activate the option – you can even mix up pictures, flash files, iFrames, FLV movies etc.

5 Displaying FLV movies using included players

It's as easy as displaying an image (see chapter 3). Two free FLV players are included with EnlargeIt!, you can chose the one you like. Here are the differences between displaying images and FLV movies:

- The included FLV players need the flash plug in. EnlargeIt! checks if the visitor has a flash plug in installed. If not, a message is displayed and nothing will be enlarged.
- To use the *rphMedia* player, the *longdesc* attribute must look like this:
  longdesc="flv::path/to/flvmovie::width::height"
  Example:
  longdesc="flv::../subfolder/mymovie.flv::450::400"
- To use the *OS FLV* player, the *longdesc* attribute must look like this:
  longdesc="fl2::path/to/flvmovie::width::height"
  Example:
  longdesc="fl2::../subfolder/mymovie.flv::450::400"
- The path to your FLV movie must be given either absolute, or relative from the directory where the EnlargeIt! files are placed on your web server; *not* from the place where your web page is.
- It is not possible to use an animation when displaying FLV movies. EnlargeIt! will never use glide or fade to display an FLV file, no matter what you set *enl_ani* to.
- You shoudn't set *enl_darksteps* too high if you want to display movies, because it can make the playback stutter on slow computers. Recommended is a value < 5.
- FLV movies do not support drag & drop.

FLV movies behave exactly like images – they can have the same buttons, they can be grouped via class attribute – you can even mix up pictures, flash files, iFrames, FLV movies etc.

6 Displaying DivX movies using DivX Web Player

It's as easy as displaying an image (see chapter 3). The DivX Web Player will be used to display the movie. This one requires the user to have some kind of browser based DivX player installed (it comes with every DivX package and is included in most Linux distributions), so you should include a link to their installer on your web page. Please note that DivX Web Player may only be embedded for non-commercial purposes. See www.divx.com for details.

Here are the differences between embedding images and DivX files:

- To embed a DivX movie, the *longdesc* attribute must look like this:
  longdesc="dvx::path/to/divxmovie::width::height"
  Examples:
  longdesc="dvx::subfolder/mymovie.divx::450::400"
  longdesc="dvx::subfolder/mymovie.avi::450::400"
- The player supports most MPEG4 files, so XVid will also work. AVI files are supported. MP3 playback will work.
- It is not possible to use an animation when displaying movies with the DivX player. EnlargeIt! will never use glide or fade to display such a file, no matter what you set *enl_ani* to.

- You shoudn't set *enl_darksteps* too high if you want to display movies, because it can make the playback stutter on slow computers. Recommended is a value < 5.
- DivX movies do not support drag & drop.

DivX movies behave exactly like images – they can have the same buttons, they can be grouped via class attribute – you can even mix up pictures, flash files, iFrames, DivX movies etc.

7 Displaying HTML content using iframes

It's as easy as displaying an image (see chapter 3). Here are the differences:

- The *longdesc* attribute must look like this:
  longdesc="ifr::URL::width::height"
  Example:
  longdesc="ifr::http://www.google.com::650::500"
- It is not possible to use an animation when displaying HTML content via iframes. EnlargeIt! will never use glide or fade to display an iframe, no matter what you set *enl_ani* to.
- Iframes do not support drag & drop.

Iframes behave exactly like images – they can have the same buttons, they can be grouped via class attribute – you can even mix up pictures, flash files, iFrames, FLV movies etc.

Internet Explorer will display its own border around the iframe. You can avoid this, if you add a CSS attribute to the source code of the page that is displayed inside the iframe. Example:

```
<body style="border:none">
```

8 Displaying YouTube movies with EnlargeIt!

It's as easy as displaying an image (see chapter 3). Here are the differences:

- EnlargeIt! checks if the visitor has a flash plug in installed. If not, a message is displayed and nothing will be enlarged.
- The *longdesc* attribute must look like this:
  longdesc="swf::http://www.youtube.com/v/**WQ1zZZ92fXo**&fs=1&rel=0::480::385"
  The bold portion must be taken from the original YouTube URL:
  http://www.youtube.com/watch?v=**WQ1zZZ92fXo**
- It is not possible to use an animation when displaying YouTube videos. EnlargeIt! will never use glide or fade to display a flash file, no matter what you set *enl_ani* to.
- You shoudn't set *enl_darksteps* too high if you want to display videos, because it can make the playback stutter on slow computers. Recommended is a value < 5.

YouTube videos behave exactly like images – they can have the same buttons, they can be grouped via class attribute, they support drag & drop if you activate the option (though it doesn't make much sense) – you can even mix up YouTube movies, other flash files, iFrames, FLV movies etc.

9 Settings in detail

Open *enlargeit.js* with a decent text editor. (BTW: *notepad.exe* is ***not*** a decent text editor, although it will work as well).

You'll see a copyright comment (that you are ***not*** allowed to remove!), followed by a number of variable definitions (each line starts with '*var*'). You may edit the values, but please make sure to not remove the semicolon (;) at the end of each line.

Setting Name	**Purpose**	**Possible values**
enl_gifpath	The path to the EnlargeIt! files (they all must be in the same folder). The last character **must** be a slash (/). For relative paths, you may start with dot-slash './'.	Examples: './subfolder/' './enlargeit/images/' 'http://mysite.com/img/' '/files/'
enl_brdsize	Sets the size of the border. Set this to 0 for no border.	A value between 0 and 20. A higher value means a bigger border.
enl_brdcolor	The colour of your border, either in hex format, or in clear text. If you leave this empty, the default will be white.	'' '#AA0000' '#FFFFFF' 'white' 'yellow'
enl_brdround	An optical gimmick, that will only work in Mozilla based browsers, Safari, Chrome and older Konqueror versions. The corners of the border will be displayed rounded. In other browsers, the corners stay square. There's no difference in terms of functionality.	0 = off 1 = on

enl_brdbck	A background picture that will be displayed across the border. Leave empty to deactivate. The graphics file must be in the folder specified by *enl_gifpath*. If you use this, *enl_brdround* will not work.	‘’ ‘b_rain.png’
enl_maxstep	The number of steps to use for animation. Will have no effect, if *enl_ani* is set to 0 (no animation).	A value between 5 and 25. Don’t set this too high, because browsers on older systems may get problems in displaying the animation, and/or the animation may take very long.
enl_speed	The time between two animation steps. The lower the value is, the faster the animation will be displayed.	A value between 10 and 25. Don’t set this too low, because browsers on older systems may get problems in displaying the animation. Don’t set this too high, because the animation may take very long then.
enl_ani	The kind of animation you want to use. For anything else than pictures (movies, iframes etc.), no animation (0) will be used, no matter what you set this to.	0 = no animation 1 = fade in/out 2 = glide 3 = bump glide 4 = smooth glide 5 = exponential glide
enl_opaglide	Only relevant for glide animation types. Combines the glide animation with a transparency effect. May perform badly in IE if many objects are on the page. Please test carefully.	0 = off 1 = on
enl_shadow	Selects if you want to use a drop shadow around the image border. Will have no effect, if enl_brd is set to 0. If you want only a shadow but no border, set *enl_brd* to 1 and *enl_brdsize* to 0.	0 = off 1 = on

enl_shadowsize	The shadow is at least 2 pixels thick. This value makes the right and bottom shadow bigger, resulting in a kind of 3D effect. Normally, you'll leave this set to 0 or 1. Will have no effect if *enl_shadow* is set to 0.	A value between 0 and 20.
enl_shadowcolor	The color of the shadow. If you leave this empty, it will be black (default). Anything else than black will look best if *enl_shadowsize* is 0.	'' 'yellow' '#ffff00'
enl_shadowintens	The shadow intensity/opacity. Will have no effect if *enl_shadow* is set to 0.	A value between 10 and 30. A higher value means a darker shadow.
enl_dark	Darken screen when a picture is enlarged. If you set this to 1, the rest of the screen will get darkened when an image is enlarged. You may know this effect from LightBox or its clones. Note that if you activate this, you can only enlarge one pic at the same time. If you want this but you don't want a dark effect, set this to 1 and *enl_darkprct* to 0.	0 = off 1 = on 2 = on and will stay dark when navigating (may result in bad performance with some animation types, please test carefully with different browsers)
enl_darkprct	How dark the screen should be darkened, can be set with this parameter. This will have no effect if *enl_dark* is set to 0.	A value between 0 and 100. 0 means no darkening at all, 100 means complete black.
enl_darksteps	How long the darkening should take. Very resource intensive, so use 1-5 if you display movies to avoid stuttering. If you set this to 1, darkening will be immediately. Always test the performance in several browsers.	1 – 30

enl_center	Normally, EnlargeIt! will try to display the pictures as close to the thumbnail as possible without leaving the viewport of your browser window. If you set this to 1, the full size pic will be centered in the middle of your browser window.	0 = off 1 = on
enl_drgdrop	If you set this to 1, users can use drag&drop with their mouse to move enlarged pics inside the browser window.	0 = off 1 = on
enl_preload	This will only be relevant if you group pictures with their class attribute and use navigation buttons. If you set this to 1, the next and previous picture of the current picture will be preloaded as soon as the current picture is enlarged. This speeds up navigation.	0 = off 1 = on
enl_titlebar	Show picture title in a title bar above the image. The title bar will stay empty if no title is given via the *alt* tag of the thumbnail. If buttons are defined, the title bar will be always displayed, no matter what you set this value to.	0 = off 1 = on
enl_keynav	If you set this to 1, visitors can use their arrow keys (left, right) to navigate, and Esc key to close the enlarged file.	0 = off 1 = on
enl_wheelnav	If you set this to 1, visitors can use their mouse wheel to navigate. I recommend using this in conjunction with darkening. Will only work in IE and Mozilla browsers.	0 = off 1 = on
enl_titletxtcol	The colour of the font that is used to display the picture title in the title bar, either in hex format, or in clear text.	'#9A9A9A' '#000000' 'red' 'yellow'

enl_ajaxcolor	The background color of AJAX snippets, either in hex format, or in clear text.	'#DDDDDD' '#000000' 'white' 'yellow'
enl_usecounter	Everytime a file gets enlarged, you can call a hidden counter page. Please see the corresponding chapter.	0 = off 1 = on
enl_counterurl	The URL of the hidden counter page.	Any relative or absolute path on the same (sub)domain.
enl_btnact	The name of the image that represents the active buttons. The file must be in the *enl_gifpath* folder.	'bact.png'
enl_btninact	The name of the image that represents the inactive buttons. The file must be in the *enl_gifpath* folder.	'binact.png'
enl_pluscur	The name of a cursor file that can be shown if you move the mouse over the thumbnail.	'pluscur.cur'
enl_minuscur	The name of a cursor file that can be shown if you move the mouse over the enlarged file.	'minuscur.cur'
enl_noflash	Text to display if one of the following file types is to be displayed, and no flash plugin is found on the visitor's computer: FLV, SWF, YouTube movie	'No flash plugin found! This file can't be displayed.'
enl_canceltext	If the full size file mentioned in *longdesc* attribute cannot be found or the web server is very slow, the AJAX loader will be shown forever. This is the tooltip that is displayed if the user moves his mouse pointer over the AJAX loader.	'Click to cancel loading of this file…'

10 Buttons

It's up to you which buttons should be visible to the users. You have to add three values to *enlargeit.js* for each button:

1. The pre-defined function or the AJAX base URL.
2. The tooltip text that is shown when the mouse pointer rests on top of a button.
3. The pixel offset in files *bact.png* and *binact.png*.

Even if you don't want to use buttons at all, the files *bact.png* and *binact.png* must be inside the folder where the EnlargeIt! files are.

There's a difference between *pre-defined buttons* and *AJAX buttons*.

10.1 Pre-defined buttons

For now there are six pre-defined actions you can use: 'close', 'next', 'prev', 'site', 'max' and 'pic'. Instead of an AJAX URL, you simply add the word. Using 'close', the image will be shrinked/closed. With 'pic' the image is made visible again, this will only make sense, if you use at least one AJAX button, so you could switch between the AJAX content and the image. The 'prev' and 'next' buttons are for navigation; when a visitor clicks on one of them, the next or previous picture of the same group will be enlarged. Grouping pictures is simple, just assign the same *class* attribute to them. See the source code of file *index.html* in Example_2 for details. The 'site' button will lead you to another page. The 'max' button loads a full size version of an intermediate sized picture.

The **close button** is defined this way:

```
enl_buttonurl[0] = 'close';
enl_buttontxt[0] = 'Close';
enl_buttonoff[0] = -126;
```

If the close button is not the first or the only button, the number in the brackets [ ] has to be increased accordingly.

If a visitor clicks into the picture when **no** close button is defined, the picture will be closed. If you have a close button, the visitor will **have to** use it to close the picture. A click into the picture will in this case only be useful for dragging the image around (drag&drop).

The **pic button** is defined this way:

```
enl_buttonurl[0] = 'pic';
enl_buttontxt[0] = 'Show picture';
enl_buttonoff[0] = -90;
```

You have to increase the number in the brackets [ ] as well if the pic button is not the first or the only button. To get a better understanding of how buttons are defined in EnlargeIt!, I recommend to study the file *enlargeit.js* you can find in the AJAX examples.

The **next and prev buttons** are defined like this:

```
enl_buttonurl[0] = 'prev';
enl_buttontxt[0] = 'Previous Picture';
enl_buttonoff[0] = -180;

enl_buttonurl[1] = 'next';
enl_buttontxt[1] = 'Next Picture';
enl_buttonoff[1] = -198;
```

The **site** button will load a new page. It is defined like this:

```
enl_buttonurl[0] = 'site:http://www.google.com';
enl_buttontxt[0] = 'Visit Google';
enl_buttonoff[0] = -72;
```

Please note that the name attribute of the thumbnail – if defined – is added to the URL; something like this would also be possible:

```
enl_buttonurl[0] = 'site:infopage.php?pictureId=';
enl_buttontxt[0] = 'Visit info page';
enl_buttonoff[0] = -72;
```

If the *name* attribute of the thumbnail is *name="80"*, the following URL will be visited:

```
infopage.php?pictureId=80
```

Last but not least, you may define a **max** button. This one will only be visible if the picture filename given by the *longdesc* attribute starts with 'normal_'.

```
enl_buttonurl[0] = 'max';
enl_buttontxt[0] = 'Maximize picture';
enl_buttonoff[0] = -252;
```

Let's assume you have three image files:

thumb_mypic.jpg	(100px x 66px)
normal_mypic.jpg	(600px x 400px)
mypic.jpg	(1800px x 1200px)

You embed the thumbnail and give it a *longdesc* attribute:

```
longdesc="normal_mypic.jpg"
```

For all image files that start with 'normal_', the visitor will get the additional max button. If it is clicked, the pic will reload with the full size version (*mypic.jpg*).

10.2 Button order

The order in which the buttons are displayed is set with the number in the brackets [ ]. The button with index [0] will be displayed as the most left button, the one with the highest number in the brackets – e. g. [3] – will be displayed as the most right button. You are not allowed to leave empty numbers, meaning that it will *not* work to define buttons with indices [0], [2] and [3] as long as no button with index [1] exists.

10.3 AJAX buttons

You can load as many AJAX snippets as you like with AJAX buttons, as long as the snippet files are hosted on a web server of the **same** (sub) domain.

Important: Loading AJAX snippets will *not* work from your local hard disk, you have to use a web server as test environment. Otherwise, EnlargeIt! will display an error message ("Ajax did not work")!

The base URL is defined with the button. Let's assume we have a document named *copyright.html* in the root folder of our web space. Now we want to dynamically load it as AJAX snippet via an info button for each picture. The button will have to be defined like this:

```
enl_buttonurl[0] = '/copyright.html';
enl_buttontxt[0] = 'Show info';
enl_buttonoff[0] = -72;
```

If an AJAX button is defined, it most of the time makes sense to also define a pic button (see above), so the visitor can switch between the AJAX page and the image. Such a definition would look like this:

```
enl_buttonurl[0] = 'pic';
enl_buttontxt[0] = 'Show image';
enl_buttonoff[0] = -90;

enl_buttonurl[1] = '/copyright.html';
enl_buttontxt[1] = 'Show info';
enl_buttonoff[1] = -72;
```

If you additionally like to define a close button and navigation buttons, the definition would look like this:

```
enl_buttonurl[0] = 'pic';
enl_buttontxt[0] = 'Show image';
enl_buttonoff[0] = -90;

enl_buttonurl[1] = '/copyright.html';
enl_buttontxt[1] = 'Show info';
enl_buttonoff[1] = -72;

enl_buttonurl[2] = 'prev';
enl_buttontxt[2] = 'Previous picture';
enl_buttonoff[2] = -180

enl_buttonurl[3] = 'next';
enl_buttontxt[3] = 'Next picture';
enl_buttonoff[3] = -198;

enl_buttonurl[4] = 'close';
enl_buttontxt[4] = 'Close';
enl_buttonoff[4] = -126;
```

10.4 What AJAX snippets are

AJAX snippets in the meaning of EnlargeIt! are short pieces of source code that are loaded as content of a <*DIV*> into your web page. They are ***not*** complete web pages (as you would use when adding an <*iframe*> to your page), they have no own <*head*> and no <*body*> tags. You cannot load own CSS or script files with AJAX snippets, they use all settings of the actual webpage they are loaded into, because the DIV is a part of it. This way, AJAX applications can be designed very fast and resource saving; most of the time, only a few bytes have to be loaded dynamically. Please keep in mind, that the width of such content is limited (to the width of the enlarged image).

10.5 Dynamic AJAX buttons

It would be quite boring to show the same description page for each image via AJAX snippet. Therefore, the real URL for the snippet is combined of

1. The value *enl_buttonurl*[*X*].
2. The *name* attribute of the image.

If your image is for example integrated into your page like this

```
<img src="images/thumb_file.jpg" id="pic1" alt="My first picture"
onclick="enlarge(this);" name=”42” longdesc="images/fullsize_file.jpg" />
```

and for your button the value would be defined like this

```
enl_buttonurl[X] = '/files/showme.php?picid=';
```

then this would be the link for the AJAX snippet loaded with this button:

/files/showme.php?picid=42

It works as well to use only the name attribute to define the URL for the AJAX snippet; in this case, you have to set *enl_buttonurl* to an empty string:

```
enl_buttonurl[X] = '';
```

10.6 Hyperlinks between AJAX snippets

Well, I didn't tell you the complete truth in the previous paragraph. The URL above is s simplified version; the real URL would look like this:

/files/showme.php?picid=42&enl_img=XXX

If you don't want to link between AJAX snippets, you can safely ignore the second value (*enl_img*) that is added by EnlargeIt! automatically. But if you want to hyperlink from an AJAX snippet to another, you have to get the parameter *enl_img* and add it to the hyperlink that leads to the next snippet. You may *not* pass the value as POST parameter.

A link from an AJAX snippet to another AJAX snippet is defined like this:

```
<a href="javascript:;" name="targetfile.php?enl_img=<?php echo $_GET['enl_img'];?>"
onclick="enl_ajaxfollow(this);"> Click here </a>
```

The target of the link is passed via the *name* attribute. The *href* attribute simply gets the value *javascript:;* and it's very important to add *onclick="enl_ajaxfollow(this);"* to the link to make sure that the right AJAX window is refreshed.

Please keep in mind: The hyperlink target (*name* attribute) value must be set including the correct path to the file, seen from the base web page, not seen from the previous AJAX snippet.

11 Hidden call of a counter URL

If you like, EnlargeIt! will do a hidden call of a counter page every time a file is enlarged. You may activate this function by setting *enl_usecounter = 1*. The call is done via AJAX, so keep in mind that the counter page has to be on the same (sub)domain. The counter page doesn't have to provide any content, because it isn't displayed anywhere.

The URL that is called is – like with AJAX buttons – build out of two components:

- The value of *enl_counterurl*.
- The *name* attribute of the thumbnail.

Example: You set

```
enl_counterurl = 'countmyviews.php?pictureId='
```

and the thumbnail has the attribute

```
name="40"
```

Then the following URL will be called:

```
countmyviews.php?pictureId=40
```

12 Enlarging a file at page load

To enlarge a file automatically, directly after the page is ready loaded, you have to do two things:

1. Give the thumbnail of the file you want to enlarge an id:
   ```
   <img src="images/thumb_file.jpg" alt="My first picture" onclick="enlarge(this);"
   longdesc="images/fullsize_file.jpg" id="myId" />
   ```
2. Set an additional parameter in your website, e. g. like this:
   ```
   <script type="text/javascript">enl_openpic="myId";</script>
   ```

13 Browser Compatibility

EnlargeIt! v1.1 works fine with any browser that was released in the last 4 years. It is tested with the following browsers and operating systems:

Internet Explorer 5.0 and above (Windows)

Mozilla Firefox 1.5 and above (Windows and Linux)

Mozilla Seamonkey 1.1 and above (Linux)

Opera 8.0 and above (Windows and Linux)

Konqueror 3.5 and above (Linux)

Safari 3 and above (Windows)

Google Chrome 1.0 and above

Most probably it will work fine with many more browsers and operating systems. If you own a Mac or use other/older browsers, please report here about your experiences: http://enlargeit.timos-welt.de/forum/

14 Customizing EnlargeIt! to your needs

EnlargeIt! is free software and open source, so you may change whatever you like. Have a look at the file LICENSE.TXT for details about the General Public License.

To keep the Javascript files as small as possible, you can use *EnlargeIt! Building Blocks* to generate new files that provide exactly the features you want to use. This way,

enlargeit_source.js can be between 21 and 47 KByte, and *enlargeit.js* can be between 11 and 22 Kbyte in size. You may visit *EnlargeIt! Building Blocks* here: http://enlargeit.timos-welt.de/english/11/building_blocks.php

If you like to change *enlargeit_source.js* to your needs, you can use an adapted version of Dean Edward's Packer to generate *enlargeit.js* out of *enlargeit_source.js* here: http://enlargeit.timos-welt.de/packer/

If you add functionality to EnlargeIt! you are invited to post your code in the forum. Please contribute to this project!

15 Do you have questions?

Please visit the support forum on the EnlargeIt! website (http://enlargeit.timos-welt.de)