

LIFT for Dreamweaver - Nielsen Norman Group Edition

Test Package: Beyond ALT Text

Nielsen Norman Group

and

UsableNet Inc.

83 Mercer Street, New York City, NY 10012

ph. 212 9655388

fax: 212 96553991

support@usablenet.com

August 28, 2002

Contents

1	Introduction	2
2	minimize graphics (2)	3
2.1	Minimize use of graphics [NNg-ACC 02]	3
3	name graphics (3)	5
3.1	Use ALT text to describe images [NNg-ACC 03]	5
3.2	Use LONGDESC to describe images in detail [NNg-ACC 03]	8
4	never blur (4)	12
4.1	Never blur pictures to indicate unavailability [NNg-ACC 04]	12
5	useful graphics (5)	14
5.1	Provide information also in text [NNg-ACC 05]	14
6	getting info (6)	16
6.1	Refer to alternate ways for getting graphics information [NNg-ACC 06]	16
7	shrunk pictures (7)	18
7.1	Avoid using shrunk-down pictures of other pages [NNg-ACC 07]	18
8	crisp images (8)	20
8.1	Use crisp and clear images [NNg-ACC 08]	20

9 skip media (9)	22
9.1 Make it easy to skip multimedia objects [NNg-ACC 09]	22
10 text only (10)	24
10.1 Avoid automatically created text-only pages [NNg-ACC 10]	24
11 pop-up windows (11)	26
11.1 Avoid opening new browser windows [NNg-ACC 11]	26
11.2 Avoid opening unrequested windows [NNg-ACC 11]	28
12 pop-up windows (14)	31
12.1 Provide easy ways to go back when a new window is opened [NNg-ACC 14]	31
12.2 Provide easy ways to get back to the site's home page [NNg-ACC 14]	34
13 cascading menus (16)	37
13.1 Avoid long cascading menus [NNg-ACC 16]	37
13.2 Avoid cascading menus [NNg-ACC 16]	38
13.3 Avoid pull-down menus [NNg-ACC 16]	39
14 links and buttons (17)	42
14.1 Avoid too many outgoing links [NNg-ACC 17]	42
15 small links (18)	44
15.1 Avoid too short text links or button labels [NNg-ACC 18]	44
15.2 Avoid tiny links [NNg-ACC 18]	45
15.3 Avoid too small buttons [NNg-ACC 18]	48
15.4 Avoid too small hotspots [NNg-ACC 18]	49
16 separate links (19)	52
16.1 Leave space between textual links [NNg-ACC 19]	52
16.2 Leave space between buttons [NNg-ACC 19]	53
16.3 Leave space between vertically aligned textual links [NNg-ACC 19]	56
17 underline links (22)	58
17.1 Underline all links [NNg-ACC 22]	58
18 links and buttons (23)	60
18.1 Avoid embedding image buttons in text [NNg-ACC 23]	60
19 confirm what the page is (25)	62
19.1 Confirm what the page is: check its title [NNg-ACC 25]	62
20 homepage logo (26)	64
20.1 Do not associate the word 'homepage' with your company logo [NNg-ACC 26]	64
21 skip links (31)	67
21.1 Use 'skip links' so users can skip links or navigational elements [NNg-ACC 31]	67

22 limit info required by forms (35)	70
22.1 Limit amount of information required by forms [NNg-ACC 35]	70
23 label close to field (36)	72
23.1 Put text label very close to field [NNg-ACC 36]	72
24 color coding (37)	74
24.1 Avoid color coding of form errors [NNg-ACC 37]	74
25 required fields (38)	76
25.1 Do not rely only on asterisk [NNg-ACC 38]	76
26 logical tab order (39)	78
26.1 Make sure tab order is logical [NNg-ACC 39]	78
27 match visual order (40)	80
27.1 Match tab order to visual layout [NNg-ACC 40]	80
28 stack fields (41)	83
28.1 Stack fields in a vertical column [NNg-ACC 41]	83
29 button close to field (43)	84
29.1 Put 'Go' button close to selection box or entry field [NNg-ACC 43]	84
30 submit button (44)	86
30.1 Put the submit button close to fields [NNg-ACC 44]	86
31 form instructions (45)	88
31.1 Put instructions before the field [NNg-ACC 45]	88
32 consider timeout (46)	90
32.1 Consider how long a timeout is [NNg-ACC 46]	90
33 small text (48)	92
33.1 Avoid small text [NNg-ACC 48]	92
34 large tables (68)	95
34.1 Avoid large tables [NNg-ACC 68]	95
35 summarize tables (71)	97
35.1 Summarize tables [NNg-ACC 71]	97
36 describe frames (73)	99
36.1 Describe all frames [NNg-ACC 73]	99
A Test Categories	102

A Test Parameters

103

1 Introduction

This package contains tests based on guidelines published by **Nielsen Norman Group** in the report "Beyond ALT Text: Making the Web Easy to Use for Users with Disabilities".

Using these guidelines will help Web developers to make easy-to-use websites for people who are blind, have low vision, or physical disabilities. The guidelines in the report and this product are based on many usability evaluations and sessions with users of assistive technology. These guidelines are different from the Section 508 guidelines and the W3C Web Content Accessibility Guidelines (WCAG) because they are based on our behavioral research studies and our usability-centered findings. We have boiled down the findings into this automated tool to help you quickly fix issues, and learn about good, usable, accessible Web design.

Priorities have been assigned to tests according to the following schema:

- **priority 1** means that users could not complete the task at all, or were significantly frustrated when this guideline was violated;
- **priority 2** means that users were very frustrated when this guideline was violated;
- and **priority 3** means that user experienced little frustration when this guideline was violated.

Guidelines and categories in this test package correspond to guidelines and their groupings used in the report. The number at the end of each guideline name is the guideline number as it appears in the "Beyond ALT Text: Making the Web Easy to Use for Users with Disabilities".

(C) 2002 Nielsen Norman Group and UsableNet

2 minimize graphics (2)

2.1 Minimize use of graphics [NNg-ACC 02]

- **Test Type:** Automatic
- **Test Categories:**
 - ALL TESTS
 - NNg - beyond ALT text
 - Images

- **Test Description:**

This test implements guideline # 2 on page 42 that says: "Minimize the use of graphics". The test counts number of images and compares them against a threshold (by default set to 15).

In particular it counts all sorts of images but spacers (i.e. transparent 1x1 images that are normally used for setting table sizes or for implementing hidden links) and images that are repeated many times in a page (like bullets).

This test has **priority 3**: user experienced little frustration when this guideline was violated.

- **Issue Description:**

Page contains too many images (generic ones, decorations, slices, image maps, banners, link images), exceeding the set threshold (by default equal to 15) stored in the parameter called "Max number of images".

- **Test Parameters:**

- **Max number of images** with default value: 15 .

This parameter can be any number greater than 0. It represents the **maximum number of images** (except spacers) that can be contained in a page.

- **How to fix:**

Reduce the number of graphics that you use in your pages. We recommend keeping them below 15.

When graphics are just extras, like a person smiling, consider first removing them from the site altogether. If they convey some meaning, consider conveying it in text instead. For example, instead of "man in red vest waving", try something like "we care about our customers".

Give **all** graphics (even advertising banners) names that are understandable and that thoroughly convey what the graphic is and does. Use ALT text to briefly describe images, and the LONGDESC attribute to thoroughly describe them.

When graphics contain useful information, also provide the information in text. This will ensure that even users who choose to turn off graphics will also have access to the information. If an image contains relevant information, use the LONGDESC attribute to thoroughly describe the image. This is obviously a good recommendation for any items that you are selling. Also, you should thoroughly describe any information in graphical timelines or hierarchical images.

- **Why to fix:**

In terms of inaccessible design, graphics are easily one of the worst offenders, and many Web developers are already well aware of this fact. The best thing for all users, whether they use assistive technology or not, is to minimize the use of graphics, increasing the speed of page load-time and decreasing superfluous noise.

People using screen readers, Braille devices, and screen magnification software complained about how site graphics impair their work. And, when they hit sites that seemed to have minimal graphics, they frequently said they liked the site, even when they had barely used it.

One important point is that, because users have encountered so many inaccessible graphics, we saw many participants turn off graphics in their browser. So, bear in mind that some users won't see your graphics at all, even if you use them sparingly.

- **Learn More:**

- HTML 4.0 standard on the [IMG tag](#);
- W3C on [how to define the ALT attribute](#);
- Alertbox on [ALT descriptions](#);
- W3C on [how to define the LONGDESC attribute](#);
- HTML 4.0 standard on the [LONGDESC attribute](#);
- another tutorial on [alternative content for graphics](#) rich of examples;
- a tutorial on the [use of ALT text in IMGs](#);

3 name graphics (3)

3.1 Use ALT text to describe images [NNG-ACC 03]

- **Test Type:** Automatic
- **Test Categories:**
 - ALL TESTS
 - NNG - beyond ALT text
 - Images
 - graphics

- **Test Description:**

This test implements guideline # 3 on page 43: "Give all graphics (even advertising banners) names that are understandable and that thoroughly convey what the graphic is and does. Use ALT text to briefly describe images, and the LONGDESC attribute to thoroughly describe them."

The test checks if the page contains images (including banners but excluding spacers, bullets and decorations) that don't have a valid ALT text.

A valid ALT text is not empty, not blank and does not contain any placeholder text (like the file name, the words "image" or "photo").

The placeholder text is specified in the parameter "ALT placeholders", by default filled with ".gif|.jpg|.jpeg|.jpe|.png|bytes". The maximum length of the ALT text is specified in the parameter "Max ALT length (chars)", by default equal to 150.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

The page contains images that are not appropriately described in text.

- **Specific Issues:**

- **image has no ALT:** The image does not have any ALT
- **ALT is empty:** ALT of image is the empty string "". (This may be OK for banners linking to pages that are known to be not accessible.)
- **ALT is blank:** ALT of image is the blank string ". (This may be OK for banners linking to pages that are known to be not accessible.)
- **ALT with html tags:** ALT of image contains HTML tags.
- **ALT too long:** ALT of image is too long.
- **ALT with placeholder text:** ALT of image appears to contain only placeholder text (specified in parameter "ALT placeholders").

- **Test Parameters:**

- **ALT placeholders** with default value: `\.gif|\.jpg|\.jpeg|\.jpe|\.png|bytes` .
This parameter can be a bar separated list of words. When an ALT of an image is analyzed, LIFT checks if the ALT contains any of these words. If so, LIFT assumes that the ALT contains only a placeholder, not significant text.
More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.
- **Max ALT length (chars)** with default value: 150 .
This parameter has to be a number. It represents the maximum number of words that may appear in the ALT attribute of an image.
More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

- **How to fix:**

HTML affords elegant means for naming images. Use the ALT attribute after the image name to briefly describe the image. Use the LONGDESC attribute to thoroughly describe the image. When writing ALT text, keep the text concise and simple. The goal is to let people using screen readers know what the image is. For example, there is probably no need to go into great detail about what the Major League Baseball logo looks like, but telling users a particular graphic is the MLB logo would help.

Some browsers may not yet support some of the HTML attributes and elements, such as LONGDESC. Our advice is to use these and allow the browsers to catch up. In cases where a page actually looks broken because of the elements, you might need some workarounds temporarily.

When using LIFT for Dreamweaver, select the image in the editor window and open the Fix Wizard. The appropriate tools will be offered to add/edit the ALT and LONGDESC attributes.

You can use the **ALT Editor** to easily manage the ALTs of your images.

Remember that:

- the description should explain the role of the image in the page. Imagine listening to the description over the phone to determine whether it is descriptive enough;
- if the image is used as the content of a link and you provide link text too, use a space as the ALT attribute value of the IMG element. In such a case link text should be the alternative description for the image;
- Be brief. Consider that images like logos are repeated on every page of the site, and people would have to listen to the same description over and over;
- As a rough guideline, for buttons use the same text that is shown by the image;
- ALT descriptions are not interpreted by browsers and should not include HTML tags. Embedded tags can only confuse users and maybe search engines as well;

- Too long ALT descriptions may be truncated by browsers and increase the time required to download the page (J. Nielsen in his book www.useit.com/jakob/webusability suggests to use less than 10 words and 64 characters);
- Images used solely for decorative purposes (called spacers) should include an empty ALT string (“”) so that they are not considered by screen-readers. Similarly for images like bullets;
- If an image is really superfluous, consider removing it altogether. If it conveys some information, be sure it is described in the ALT text. Consider whether the ALT text should simply describe the image, i.e. “man waving” or instead give some better information, such as “we care about our customers”.
- **BEWARE:** the ALT attribute should be the empty string (“”) in cases where the image is already described by surrounding text;
- **BEWARE:** all images included in A links (including transparent GIFs) need to have valid ALTs describing the link destination.

- **Why to fix:**

For screen reader users, there are few things more frustrating than waiting for a screen reader to read something meaningless. It’s like waiting in a long line only to get to the front and find out the person behind the counter cannot help you.

The only indication people using screen readers have of what the image is is what the developer explicitly tells them. We saw many examples of poorly named images, poorly used ALT text attributes, or images with no ALT text at all.

A financial site used an image to convey the risk level of a particular mutual fund. The images were named Risk 1 through Risk 5, and they had no ALT text associated with them. The image itself did not use a numeric risk level, but rather indicated a high, medium, or low risk level. It would have helped screen reader users to have the images named High Risk, Low Risk, etc., or to have the ALT text provide this information.

If an image contains relevant information, use the LONGDESC attribute to thoroughly describe the image. This is obviously a good recommendation for any items that you are selling. Also, you should thoroughly describe any information in graphical timelines or hierarchical images. On an e-commerce site, several buttons were not properly labeled. The screen reader read only the words “Push button”. Sighted users would see these words on the buttons:

- Change amount (white button)
- See my shopping bag (light blue button)
- Check out (red button)
- Continue shopping (navy blue button)

In another example, there was no clear difference between the buttons that appear under each item when read by a screen reader. The words on the buttons, as seen by a sighted user, are: “Description of product” and “Shopping”. When the screen reader read them, it said: “Go to product_detail”, and “Go to product_select”.

- **Learn More:**

- W3C on [how to include objects and images](#);
- HTML 4.0 standard on the [IMG tag](#);
- a tutorial on the [use of ALT text in IMGs](#);
- another tutorial on [alternative content for graphics](#) rich of examples;
- Alertbox on [ALT descriptions](#).

3.2 Use LONGDESC to describe images in detail [NNg-ACC 03]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- Manual
- NNg - beyond ALT text
- Images
- graphics

- **Test Description:**

This test implements guideline # 3 on page 43: "Give all graphics (even advertising banners) names that are understandable and that thoroughly convey what the graphic is and does. Use ALT text to briefly describe images, and the LONGDESC attribute to thoroughly describe them."

The test checks if the page contains images (including banners but excluding spacers, bullets and decorations) and asks the user to verify that the LONGDESC attribute is needed.

If there is already a LONGDESC attribute defined, the test checks if it is valid (i.e. if the file exists, if the file contains HTML, if the file can be accessed).

This test excludes images whose size is smaller than the value specified by parameter "Min. width/height of significant images" (whose default is 50).

The test is manual because there is no effective way to automatically determine if the image needs a long description.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

The page contains images that appear to need a long description.

- **Specific Issues:**

- **LONGDESC attribute not defined:** Missing both the LONGDESC attribute and the D-link element.
- **LONGDESC file does not exist:** Invalid LONGDESC attribute: mentioned file does not exist.
- **LONGDESC file is not HTML:** Invalid LONGDESC attribute: mentioned file is not an HTML file.
- **LONGDESC file is empty:** Invalid LONGDESC attribute: no file specified.
- **LONGDESC URL is dead:** Invalid LONGDESC attribute: its URL is dead.
- **LONGDESC URL generated bad response:** Invalid LONGDESC attribute: got an error response from the server of its URL.
- **LONGDESC URL has a bad protocol:** Invalid LONGDESC attribute: it is not a local file nor an HTTP URL.

- **Test Parameters:**

- **Min. width/height of significant images** with default value: 50 .
This parameter represents the minimum width or height that an image has to have in order for LIFT to consider it an image that can convey information.

- **How to fix:**

The image appears to convey information and it may require a detailed and full description. If the image is a product picture, a diagram, a histogram or a chart this is the case. Please check if the image requires a long description. If not, skip this issue.

The ALT attribute cannot be used to provide a full length description of the content of an image. To describe its content you need more flexibility.

The **LONGDESC** attribute can be used to provide a long description of the associated image that would not fit in the ALT attribute. By including **LONGDESC="any_HTML_file"** in your IMG tag you can link the image to the HTML file containing a formatted description of the image. The long description (unlike the ALT attribute) can contain HTML code, with links to other resources, formatting instructions, etc.

When using LIFT for Dreamweaver, select the image in the editor window and open the Fix Wizard. The appropriate tools will be offered to add/edit the ALT and LONGDESC attributes.

Please define a proper **LONGDESC** attribute for the image (by specifying the address – URL – of an HTML file).

Some browsers may not yet support the LONGDESC attribute. Our advice is to use these and allow the browsers to catch up. In cases where a page actually looks broken because of the elements, you might need some workarounds temporarily.

Alternatively, placing a rich textual description close to the image (like a caption) is a viable solution. In such a case you don't need the **LONGDESC** attribute.

- **Why to fix:**

The only indication people using screen readers have of what the image is is what the developer explicitly tells them. We saw many examples of poorly named images, poorly used ALT text attributes, or images with no ALT text at all.

For screen reader users, there are few things more frustrating than waiting for a screen reader to read something meaningless. It's like waiting in a long line only to get to the front and find out the person behind the counter cannot help you.

Also, many sites show images to convey a feeling, idea, or emotion, but the descriptions do not also convey this information. Just creating a short ALT tag is often not enough to convey the feeling or information a picture might give. Consider not just describing the exact image for the users who are working with screen readers. Also consider rewriting the image's meaning in text so people who are blind will also get the feeling the image sends to sighted users.

A financial site used an image to convey the risk level of a particular mutual fund. The images were named Risk 1 through Risk 5, and they had no ALT text associated with them. The image itself did not use a numeric risk level, but rather indicated a high, medium, or low risk level. It would have helped screen reader users to have the images named High Risk, Low Risk, etc., or to have the ALT text provide this information.

If an image contains relevant information, use the LONGDESC attribute to thoroughly describe the image. This is obviously a good recommendation for any items that you are selling. Also, you should thoroughly describe any information in graphical timelines or hierarchical images. On an e-commerce site we tested, several buttons were not properly labeled. The screen reader read only the words "Push button". Sighted users would see these words on the buttons:

- Change amount (white button)
- See my shopping bag (light blue button)
- Check out (red button)
- Continue shopping (navy blue button)

In another example, there was no clear difference between the buttons that appear under each item when read by a screen reader. The words on the buttons, as seen by a sighted user, are: "Description of product" and "Shopping". When the screen reader read them, it said: "Go to product_detail", and "Go to product_select".

- **Learn More:**

- HTML 4.0 standard on the [IMG tag](#);
- HTML 4.0 standard on the [LONGDESC attribute](#);
- W3C on [how to define the LONGDESC attribute](#);
- W3C on [how to include objects and images](#);

- HTML 4.0 standard on the [IMG](#) tag;
- some examples about [LONGDESC](#) and [D-links](#);
- Alertbox on [ALT](#) descriptions.

4 never blur (4)

4.1 Never blur pictures to indicate unavailability [NNg-ACC 04]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- Manual
- NNg - beyond ALT text
- Links
- Images
- graphics
- generic guidelines

- **Test Description:**

This test implements guideline # 4 on page 46: "Never blur pictures to indicate unavailability".

This test flags pages that contain navigation bars with one or more images that are disabled. That is, that cannot be clicked on.

The test does not examine image maps (aka hotspots). It only considers images embedded in A links within navigation bars (usually tables).

The test is manual because there is no effective way to detect whether:

- an image is actually the icon for a link (the test will sometimes flag images that are used to separate links)
- an image is blurred.

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

The page contains an image that is embedded in a navigation bar (usually a table) and that looks like being a disabled button.

Avoid using a blurred image to indicate that the button is not available.

- **How to fix:**

Please check that the image is not blurred.

If it is, then consider using a different but crisp image to indicate that the option is not available.

- **Why to fix:**

When graphics are blurred, it makes it even more difficult for people with low vision to decipher the picture. And, since the screen is zoomed in so closely, the context provided by other unblurred graphics is often lost.

On one of our study, a website blurred some graphics to indicate unavailability: when the Previous and Next buttons were inactive, the pictures were blurred.

A better idea is to remove graphics altogether when they are not available.

- **Learn More:**

- HTML 4.0 standard on the [IMG tag](#)

5 useful graphics (5)

5.1 Provide information also in text [NNg-ACC 05]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- Manual
- NNg - beyond ALT text
- Links
- Images
- graphics
- generic guidelines

- **Test Description:**

This test implements guideline # 5 on page 47: "When graphics contain useful information, also provide the information in text".

This test flags pages with images that may contain information useful to screen reader users. The test asks the user to check if this is the case, and if so it suggests to adequately describe the image.

This test excludes images whose size is smaller than the value specified by parameter "Min. width/height of significant images" (whose default is 50).

The test is manual because there is no effective way to automatically determine if the image contains useful information.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

The page may contain an image with information useful to screen reader users that may not be adequately described using text.

- **Test Parameters:**

- **Min. width/height of significant images** with default value: 50 .

This parameter represents the minimum width or height that an image has to have in order for LIFT to consider it an image that can convey information.

- **How to fix:**

Please check if the image contains information that might be useful to screen reader users. If so, make sure that the image is described with text either in the page or in its ALT or LONGDESC.

Use the ALT attribute after the image name to briefly describe the image. Use the LONGDESC attribute to thoroughly describe the image.

When writing ALT text, keep the text concise and simple. The goal is to let people using screen readers know what the image is. For example, there is probably no need to go into great detail about what the Major League Baseball logo looks like, but telling users a particular graphic is the MLB logo would help.

Conversely, a financial website used an image to convey the risk level of a particular mutual fund. The images were named Risk 1 through Risk 5, and they had no ALT text associated with them. The image itself did not use a numeric risk level, but rather indicated a high, medium, or low risk level. It would have helped screen reader users to have the images named High Risk, Low Risk, etc., or to have the ALT text provide this information.

- **Why to fix:**

Some graphics are just stock art or logos, and even if described, a person who is blind or person with low vision might not get much information from them. There are times, however, when graphics comprise useful information. When this is the case, users should be able to access it in HTML.

On the Rock and Roll Hall of Fame website, one screen reader and Braille user found a timeline of rock-and-roll related events. She initially thought she would be unable to read it; she was surprised and glad to find the timeline available in HTML.

For screen reader users, there are few things more frustrating than waiting for a screen reader to read something meaningless. It's like waiting in a long line only to get to the front and find out the person behind the counter cannot help you. The only indication people using screen readers have of what the image is is what the developer explicitly tells them.

We saw many examples of poorly named images, poorly used ALT text attributes, or images with no ALT text at all. HTML affords elegant means for naming images. Use the ALT attribute after the image name to briefly describe the image. Use the LONGDESC attribute to thoroughly describe the image.

- **Learn More:**

- HTML 4.0 standard on the [IMG tag](#);
- HTML 4.0 standard on the [LONGDESC attribute](#);
- W3C on [how to define the LONGDESC attribute](#);
- W3C on [how to include objects and images](#);
- Alertbox on [ALT descriptions](#);
- some examples about [LONGDESC and D-links](#).

6 getting info (6)

6.1 Refer to alternate ways for getting graphics information [NNg-ACC 06]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- Manual
- NNg - beyond ALT text
- Images
- graphics
- generic guidelines

- **Test Description:**

This test implements guideline # 6 on page 49: "Refer users to alternate ways for getting information contained in any graphics they encounter".

This test flags pages with images that may represent complex pictures and that has no LONGDESC attribute defined. The test asks the user to check if the picture is adequately described in the page, or if an adequate description can be reached by following a link from the page.

This test excludes images whose size is smaller than the value specified by parameter "Min. width/height of significant images" (whose default is 50).

The test is manual because there is no effective way to automatically determine if the image is adequately described.

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

The page contains an image that may represent a complex picture and that has no LONGDESC attribute defined. If the image requires a detailed description and if such a description is not given in the page itself, then there should be at least some link in the page pointing to such a description.

- **Test Parameters:**

- **Min. width/height of significant images** with default value: 50 .
This parameter represents the minimum width or height that an image has to have in order for LIFT to consider it an image that can convey information.

- **How to fix:**

Please check if the image requires a detailed description.

If so, provide an alternative detailed description of the graphics in the text next to the image itself or provide a link to a page that describes it.

The **LONGDESC** attribute can be used to provide a long description of the associated image that would not fit in the ALT attribute. By including **LONGDESC="any_HTML_file"** in your IMG tag you can link the image to the HTML file containing a formatted description of the image. The long description (unlike the ALT attribute) can contain HTML code, with links to other resources, formatting instructions, etc.

However, since not many browsers at the moment support the **LONGDESC** attribute, you may also want to place, near the image, a so called D-link, a normal textual link with label "D" pointing to an HTML page providing a full image description. Example:

```
<IMG src="chart.gif" alt="chart of beverages distribution"
longdesc="chart.html"><A href="chart.html">D</A>
```

- **Why to fix:**

Ideally, information in graphics and information in text will be tightly linked together. Until that ideal can be reached, some sites provide information both graphically and in text. The problem is that sometimes users find only the graphic, and are not aware that the alternate source exists.

During one of our studies, on a city website there is a very informative table of graphics that tells people which items they can recycle. This is very helpful for people who are sighted. But for people with low vision, it is extremely difficult to use. One person using a screen magnifier spent about 15 minutes trying to use it and said "This is impossible for me. I can't use this."

All the information in the graphic is also available in text form on a completely different section of the site, but most users did not find it. Because they found only the graphic, they assumed that it was the only source for the information they needed.

A better design would show the text next to the graphic, or link to the text, so it is easier to find.

- **Learn More:**

- [HTML 4.0 standard on the IMG tag](#)
- [W3C on how to define the ALT attribute](#)
- [HTML 4.0 standard on the LONGDESC attribute](#)
- [W3C on how to define the LONGDESC attribute](#)
- some examples about [LONGDESC and D-links](#)

7 shrunk pictures (7)

7.1 Avoid using shrunk-down pictures of other pages [NNg-ACC 07]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- Manual
- NNg - beyond ALT text
- Links
- Images
- graphics
- generic guidelines

- **Test Description:**

This test implements guideline # 7 on page 50: "Do not shrink down a picture of an page on your site and use it as graphic (or button) on another page".

This test flags pages with images that may represent pictures of other pages (or part of pages). The test asks the user to check if this is the case, and if so it suggests to replace them.

This test excludes images whose size is smaller than the value specified by parameter "Min. width/height of significant images" (whose default is 50).

The test is manual because there is no effective way to automatically determine if the image represents a shrunk-down picture.

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

The page contains an image that may represent a picture of other page of the site (or part of pages). If this is the case you should avoid to shrink down a picture of an actual page of the site and use it as a graphic (or button) on another page.

- **Specific Issues:**

- **thumbnail image used:** The image is a thumbnail (that is, a link to another image). It may be a shrunk-down picture of the larger image.
- **generic image used:** The image may represent a shrunk-down picture of an actual page of the site.

- **Test Parameters:**

- **Min. width/height of significant images** with default value: 50 .

This parameter represents the minimum width or height that an image has to have in order for LIFT to consider it an image that can convey information.

- **How to fix:**

Please check if the image represents a shrunk-down picture of another page (or part of the page) or if it is a thumbnail image (that is, an image that links to a larger image).

If so, replace the shrunk-down picture with something else: use a different image or simply a textual link.

- **Why to fix:**

Because screen magnifier users zoom in to view pages, small images can appear extremely large. During our study, we found websites where images of reduced pages were used as navigation buttons. When you see the whole page at once, the relative sizes make this use perfectly clear. But at 6x magnification, for example, those thumbnail page images look like real pages.

The button users needed to click to access a specific form on a website was actually a small picture of the form itself. The graphic was blurry, so people using a screen magnifier zoomed in. When they did this, they thought they were looking at the actual form and tried to fill it in.

Another site also showed a small picture of an inactive tool. Users with low vision had magnified the tool so it looked real, and they tried to use it. One participant with low vision who had been clicking the arrows said, "Maybe this would be more meaningful to someone more familiar with mutual funds. Looks like they give you more info if you use these buttons to scroll back and forth. I'm reluctant to spend time fiddling with this because I'm not sure it has what I'm looking for. They seem to give some info about returns, but they don't seem to say what the content is, what kind of investment it is. I'm getting the impression that this perhaps is not their full list of funds, but just the funds they're promoting. Maybe what I'm looking for is someplace else. If I click on these samples, nothing's changing. It's not letting me select what I want from the chart". Another user had the same problem and said, "OK, now I'm pretty much lost. I'm not getting anything".

- **Learn More:**

- [HTML 4.0 standard on the IMG tag](#)

8 crisp images (8)

8.1 Use crisp and clear images [NNg-ACC 08]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- Manual
- NNg - beyond ALT text
- Links
- Images
- graphics
- generic guidelines

- **Test Description:**

This test implements guideline # 8 on page 50: "When you do use graphics, always choose crisp and clear images".

This test flags pages with images that may be fuzzy. The test considers all sorts of images except for spacers, decorations, and bullet-like images.

This test excludes images whose size is smaller than the value specified by parameter "Min. width/height of significant images" (whose default is 50).

The test is manual because there is no effective way to automatically determine if the image is fuzzy.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

The page contains an image that may be non crisp or non clear. You should always choose crisp and clear images, especially if they contain text.

- **Test Parameters:**

- **Min. width/height of significant images** with default value: 50 .

This parameter represents the minimum width or height that an image has to have in order for LIFT to consider it an image that can convey information.

- **How to fix:**

Please check if the image is crisp and clear. Try to magnify, or zoom in, the image and see if its contents can be clearly perceived and read.

If no, replace the image with another one that is crisper and clearer.

- **Why to fix:**

People with low vision have trouble telling what some blurry graphics are, even when they magnify them. Pictures of text pose problems as well, because the letters tend to get blurrier as they are magnified.

When one participant with low vision encountered a blurry graphic, he said, "Do I have to download something? If there's print here, I can't read it at all. Give me some information here. You're pretty, but: Is this something I'm supposed to download? It's not clear at all."

- **Learn More:**

- HTML 4.0 standard on the [IMG tag](#)

9 skip media (9)

9.1 Make it easy to skip multimedia objects [NNg-ACC 09]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- graphics
- Links
- multimedia

- **Test Description:**

This test implements guideline # 9 on page 53: "Make it easy for users to skip any multimedia and Flash demos".

This test flags objects that have multimedia content that cannot be skipped by a screen reader user.

The test checks that the object (either OBJECTS, EMBED, or APPLET) is directly preceded by a **hidden link** and possibly followed by a **named anchor**.

- **Issue Description:**

The page contains a multimedia object (OBJECT, EMBED, APPLET) for which there is no hidden link that screen reader users can follow to skip the object.

- **Specific Issues:**

- **Unskippable multimedia object with sound:** The object appears not to be skippable by screen reader users. It also appears to contain sound. Make sure that either it can be skipped or that its sound effects can be turned off.
- **Unskippable multimedia object:** The object appears not to be skippable by screen reader users. Make sure that it can be skipped.

- **How to fix:**

Make it easy for screen reader users to skip any multimedia object.

One effective technique is using a **hidden link**. Take a spacer image, wrap it within an IMG tag and add an appropriate ALT text (like "skip Flash animation"). Then include the IMG in an A link and let it point to a named anchor that is added just after the object. Finally put the A just before the object.

This would be the final HTML fragment:

```
<a href="#skipF">
  
</a>
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
  codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#vers
  width="36" height="36">
  <param name="movie" value="../flash_objects/Biking.swf">
  <param name="quality" value="high">
  <embed src="../flash_objects/Biking.swf"
    width="36" height="36" quality="high"
    pluginspage="http://www.macromedia.com/go/getflashplayer"
    type="application/x-shockwave-flash">
  </embed>
</object>
<a name="skipF"></a>
```

You can use the Fix Wizard to modify the HTML code automatically.

- **Why to fix:**

Some screen readers simply cannot read Flash demos or other multimedia (or Java applications). Also, when any sounds play on a site, screen reader users often need to mute them in order to hear the screen reader.

Make it very easy for screen reader users to avoid any multimedia altogether.

- **Learn More:**

- Netscape on the [EMBED](#) element;
- W3C on [how to include objects and images](#);
- W3C on [Generic inclusion: the OBJECT element](#);
- W3C Accessibility guidelines on [multimedia equivalent alternatives](#).
- a tutorial on [accessible navigation](#)

10 text only (10)

10.1 Avoid automatically created text-only pages [NNg-ACC 10]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- Links
- graphics

- **Test Description:**

This test implements guideline # 10 on page 53: "Do not automatically create a text-only version of your site".

This test flags pages that contain links to text-only pages.

In particular the test looks for textual links, buttons, hidden links (i.e. an A link containing a spacer image) labeled with words like "text only". These words are taken from the value of the parameter "Pattern for 'text only' links".

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

The page links to a text-only version of it. If you can, avoid using text-only versions of your pages: users usually assume that text-only sites are seldom updated and less worth a visit.

- **Specific Issues:**

- **Link to text-only version:** The page has a normal A link that points to a text-only version of the page.
- **Hidden link to text-only version:** The page has a hidden link (i.e. an A link containing a spacer image) that points to a text-only version of the page.
- **Button to text-only version:** The page has a button that points to a text-only version of the page.

- **Test Parameters:**

- **Pattern for 'text only' links** with default value: `(text\s*only)|(only\s*text)|(text\s*vers`

.
This parameter represents the content of textual links or of ALT of images embedded within links. When this content pattern is found within these elements, the link is assumed to lead to a text-only version of the page.

More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

- **How to fix:**

Unless absolutely necessary, avoid using a text only version of your pages.

If you have to use it, keep in mind that even in a text-only version, accessibility issues can still occur. For example, fields and buttons still appear and must be attended to. And in any case, the homepage users hit before they can choose text-only must be accessible.

In any case do not create **automatically** the text-only version of your pages. The result is very likely to be inadequate.

- **Why to fix:**

People using screen readers did use the text-only versions of sites when they were available, as did a few people using screen magnifiers. Offering a text-only version does not release designers from making the graphical version accessible, however. Knowing that graphical sites should be accessible, organizations must choose the best use of their resources; creating and maintaining two site versions is costly and time-consuming.

The best candidates for additional text-only sites are those companies that are attempting to use images or multimedia to portray a certain impression, feeling, sensation, or emotion with their site. If the "cool factor" is your business, do what is right for your company, and also provide a text-only version of the site. On this version, strive to describe in text the messages you are otherwise sending through multimedia.

During our study we found sites with a text-only version that gives the same information as the graphical version. The homepage also reads several graphics and other information that is not understandable before users get the text-only option. Also there were often buttons and form fields that users must be able to understand, even on text only pages.

- **Learn More:**

- Alertbox on [text only sites](#)

11 pop-up windows (11)

11.1 Avoid opening new browser windows [NNg-ACC 11]

- **Test Type:** Automatic
- **Test Categories:**
 - ALL TESTS
 - NNg - beyond ALT text
 - pop-up windows, new windows
 - Links
- **Test Description:**

This test is an implementation of the guideline # 11 on page 57 that says:

Avoid opening new browser windows.

The test checks if the page opens a new browser window. It checks if the target attribute of links (A, AREA, FORM) is "_blank" or "NAME" (where NAME is defined by the page author) so that the link opens a new window.

It also checks if the link contains code like 'javascript:window.open()' that opens a new window. Or if there is code for handling ONCLICK and ONKEYPRESS events that opens a new window by executing window.open().

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

Page contains links that open new browser windows (either by specifying a target attribute, or by invoking javascript:window.open() or by invoking window.open() in the code that handles the "ONCLICK" or "ONKEYPRESS" events).

- **How to fix:**

Use standard web pages instead of opening new pop-up windows. And ensure the Back button will work in every situation, including any pop-up windows.

If the target attribute has been used, remove it from the link, form or hotspot (unless the link is part of a framed page and the target attribute refers to a frame).

You can use the Fix Wizard to do this automatically for you.

If links with href="javascript:open()" are used, then substitute them with links opening normal pages (within the currently used browser window).

Consider also that these links will not work on browsers where javascript is not enabled (for example, many organizations disable javascript on browsers for security reasons).

If there are event handlers that open new windows, then you should at least make sure that they do not open a new window. (In most cases you can change one of the arguments of the script that is invoked as a handler for the "ONCLICK" or "ONKEYPRESS" events.)

Also these links will not work on browsers where javascript is not enabled.

- **Why to fix:**

Opening new browser windows is confusing to people who use screen readers and screen magnifiers. They often do not know another window has opened in front of the primary one, so they never get to the information in the windows behind. And they sometimes click the "Back" button but nothing happens. More often than not, it seems there is a bug with the website, and they close and reopen the browser.

Pop-up windows were disorienting and often bewildering to people using screen readers and magnifiers, and often to people with motor skill disabilities as well. For people using screen readers, it can be a big problem if they click something, expect a certain page, and then hear something completely foreign like the contents of an unexpected pop-up window. The information they want, expect, and need is often right behind the pop-up window, but since the window focuses on the pop-up and that's what the screen reader reads, the user has no idea that what they want is right there.

For people using screen magnifiers, having a sense of context of every open window can be very difficult, since they only see small parts of the screen at once.

In our testing, we saw many examples where pop up windows impaired the users context and process. For example, one participant noted: "You press a link and you end up someplace way far from where you were and Back won't work. The only thing you can do is close the window." It is important to note that for users with screen readers or magnifiers, even a very small pop-up window eclipses the pages behind it.

For example, a person who is blind might go to a large city's transit authority's website to figure out how to take public transportation. Many transportation sites offer a "Trip Planner" feature that allows users type in their starting street and their destination. They can also choose arrival and departure times. On one site we tested, when the user clicked the "Trip Planner" graphic link, the site took them to the Trip Planner form. However, there was a pop-up window in front of it that was selected. Although it's a small window, for people using screen readers or magnifiers, it effectively eclipses the page behind it. In one instance, the screen reader read "Trip Planner" but the screen focus was still in the pop-up window. The user typed in her information, but the focus was not in a text box. She was lost and said, "I can't make heads or tails of this. I feel like such an idiot." On this same site, when a participant found the Trip Planner window after missing it several times because of the pop-up, she asked, "Why didn't I see this before? How did this appear? I didn't even see where it appeared as an option. I would have gone here first."

- **Learn More:**

- HTML 4.0 standard on [TARGET attribute](#)
- [tutorial on javascript windows](#)
- [tutorial on opening new windows with javascript](#)
- HTML 4.0 standard on [events](#)
- W3C on how to use [scripts](#)
- W3C/WAI techniques for making [accessible SCRIPTS](#) elements

- W3C on how to use the [NOSCRIPT element](#)
- HTML 4.0 standard on the [SCRIPT tag](#)
- HTML 4.0 standard on [links in general](#)

11.2 Avoid opening unrequested windows [NNg-ACC 11]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- pop-up windows, new windows
- Scripts

- **Test Description:**

This test implements guideline # 11 on page 57 that says: "Avoid using pop-up or new windows".

The test checks if the page contains a script that opens a new window when page is loaded (pop-up windows). More specifically it checks the code contained in the ONLOAD attribute of BODY tag and all the JavaScript instructions inside the document that are not part of a function.

If this code contains or calls functions containing `window.open()`, the test signals an automatic issue.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

Page contains javascript code that appears to open new browser windows when page is loaded (pop-up windows).

- **Specific Issues:**

- **Scripts open new windows:** Scripts inside the page contain or call functions that contain `window.open()` instruction.
- **ONLOAD attribute of BODY opens new windows:** The ONLOAD attribute of BODY tag contains a `window.open()` instruction.

- **How to fix:**

The page contains scripts (within the SCRIPT tag or inside the ONLOAD attribute of BODY tag) that open new windows when page is loaded (pop-up windows). This is achieved within the script by an instruction called **window.open**.

Change the script to avoid opening new windows. The best thing to do is present information on a standard web page and provide easy ways to navigate back and forth between pages.

If the purpose of opening a new window was to provide more context to the user while reading the new page, you might consider using frames to achieve that.

But frames bring their own accessibility issues. Some screen readers don't support frames well. For example, screen reader users are faced with the same issues with frames as they are with pop-up windows and new browser windows - it appears that the selected frame is the only available option on the page. Other frames on the page are inaccessible.

- **Why to fix:**

Like opening new browser windows, opening pop-up windows is especially confusing to people using screen readers. The screen reader (or Braille device) reads the open pop-up window text, but not the browser window behind it. In fact, users do not know there is a window behind it.

Pop-up windows were disorienting and often bewildering to people using screen readers and magnifiers, and often to people with motor skill disabilities as well. For people using screen readers, it can be a big problem if they click something, expect a certain page, and then hear something completely foreign like the contents of an unexpected pop-up window. The information they want, expect, and need is often right behind the pop-up window, but since the window focuses on the pop-up and that's what the screen reader reads, the user has no idea that what they want is right there.

One participant noted: "You press a link and you end up someplace way far from where you were and Back won't work. The only thing you can do is close the window." It is important to note that for users with screen readers or magnifiers, even a very small pop-up window eclipses the pages behind it. For example, a person who is blind might go to a large city's transit authority's website to figure out how to take public transportation. Many transportation sites offer a "Trip Planner" feature that allows users type in their starting street and their destination. They can also choose arrival and departure times. On one site we tested, when the user clicked the "Trip Planner" graphic link, the site took them to the Trip Planner form. However, there was a pop-up window in front of it that was selected. Although it's a small window, for people using screen readers or magnifiers, it effectively eclipses the page behind it. In one instance, the screen reader read "Trip Planner" but the screen focus was still in the pop-up window. The user typed in her information, but the focus was not in a text box. She was lost and said, "I can't make heads or tails of this. I feel like such an idiot." On this same site, when a participant found the Trip Planner window after missing it several times because of the pop-up, she asked, "Why didn't I see this before? How did this appear? I didn't even see where it appeared as an option. I would have gone here first."

- **Learn More:**

- HTML 4.0 standard on [TARGET attribute](#)
- [tutorial on javascript windows](#)

- [tutorial on opening new windows with javascript](#)
- [HTML 4.0 standard on events](#)
- [W3C on how to use scripts](#)
- [W3C/WAI techniques for making accessible SCRIPTS elements](#)
- [W3C on how to use the NOSCRIPT element](#)
- [HTML 4.0 standard on theSCRIPT tag](#)
- [HTML 4.0 standard on links in general](#)
- [HTML 4.0 standard on frames](#)
- [a discussion on creating quality frames](#)
- [J. Nielsen's Alertbox on frames](#)

12 pop-up windows (14)

12.1 Provide easy ways to go back when a new window is opened [NNG-ACC 14]

- **Test Type:** Automatic
- **Test Categories:**
 - ALL TESTS
 - NNG - beyond ALT text
 - pop-up windows, new windows
 - Links
- **Test Description:**

This test is an implementation of the guideline # 14 on page 60 that says "If you do open new browser windows, always provide a simple way to get back to the site's main homepage".

In particular the test looks if the the current document has links that open a new window. In particular it looks for javascript code associated to events ONLOAD, ONCLICK, ONKEYPRESS or directly specified in the HREF attribute of the link that executes the instruction `window.open()`.

If the page contains such a code, then the test scans the page to be opened looks if that page can be easily closed (or if the user can easily go back from it).

In particular, the test looks for the following things:

- if there is a link with a label matching the words specified in the parameter called "Window return label"
- if there is a link whose HREF attribute, or event handlers for ONCLICK and ONKEYPRESS events, invokes the function `window.close()`
- if there is a link whose HREF attribute, or event handlers for ONCLICK and ONKEYPRESS events, executes an instruction like "`window.location.href = previous`", where "previous" is a javascript variable that is set to `window.opener.location.href` (in the ONLOAD event of the BODY element of the page).

The test checks first if the page opens new windows. Then it looks for link or button labels (in the new window) containing the words specified in parameter "Window return label" (by default "close|return|previous") or for links and scripts that contain the code we suggest as a solution. If neither is found, then an issue is raised.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

The page contains links/buttons that open a page on a new window. It appears that from this page there is no easy way to go back (either by closing it or by returning to previous page).

- **Test Parameters:**

- **Window return label** with default value: `close|return|previous` .

This parameter can be a bar separated list of words. It represents the **possible words that can be labels of a link or button** that closes a window or that returns to previous page. (Words are case-insensitive.)

More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

- **How to fix:**

Do what you can to avoid opening pop-up windows.

If you have to have them, make sure the "BACK" button works in all cases. (Think of it as a combination of "undo" and a history list.) People depend on "BACK".

In cases where "BACK" does not work, it is important for people with no vision and low vision to be able to find a link to the homepage, so they can start over from there. For people with low vision, it would help if this link was in the upper-left corner, part of the company logo. For screen reader users, it should be one of the first five things read on a page.

If, when following a link, a pop-up window opens at the same time there is also a page refresh, make sure the page remains in the foreground and is the default window, not the pop-up.

We suggest the following solution based on javascript (you can use the Fix Wizard to do this automatically for you):

- To open a new page in a new window, use the `window.open()` javascript function (within an event handler for `ONCLICK` or `ONKEYPRESS` events, or within the `HREF` attribute of a link).
- When the new page is loaded a script stores the URL of the previous page in a variable.
- In the new page there is a link with javascript associated to it that opens the previous page in the newly opened window.

More specifically:

Within the `BODY` element of **the page to be opened** in the new window, define the following:

```
<BODY ONLOAD="un_setPrevious()">
```

and within the HEAD element add the following script (the "un_previous()" function is invoked when the page is loaded by the browser and it simply stores in a variable the URL of the previous page).

```
<script type="text/javascript">
<!--
var un_previous = '#';
function un_setPrevious()
{
    if (window.opener && !window.opener.closed)
        un_previous = window.opener.location.href;
}
//-->
</script>
```

The new page should contain a "return" link (to open the previous page) coded like the following example:

```
<a href="#" onclick="window.location.href = un_previous; return false;">
previous page</a>
```

In this way the new page handles all the information about the previous URL, and is able to open it when the link is pressed.

- **Why to fix:**

We saw many cases when people using screen readers did not realize that several browser windows were open at once. If a site automatically opened another window, they tried clicking the "Back" button to get home. If "Back" failed them, people looked for a link to get to the homepage.

Pop-up windows were disorienting and often bewildering to people using screen readers and magnifiers, and often to people with motor skill disabilities as well. For people using screen readers, it can be a big problem if they click something, expect a certain page, and then hear something completely foreign like the contents of an unexpected pop-up window. The information they want, expect, and need is often right behind the pop-up window, but since the window focuses on the pop-up and that's what the screen reader reads, the user has no idea that what they want is right there.

One participant noted: "You press a link and you end up someplace way far from where you were and Back won't work. The only thing you can do is close the window." It is important to note that for users with screen readers or magnifiers, even a very small pop-up window

eclipses the pages behind it. Some sites open the expected page a link leads to, but also open another pop-up window in front of it. A person using a screen reader or even a screen magnifier has no way of knowing that the page they want is open, but behind the pop-up window.

- **Learn More:**

- [tutorial on javascript windows](#)
- [tutorial on opening new windows with javascript](#)
- W3C on how to use [scripts](#)
- W3C/WAI techniques for making [accessible SCRIPTS](#) elements
- HTML 4.0 standard on the [SCRIPT](#) tag
- HTML 4.0 standard on [events](#)

12.2 Provide easy ways to get back to the site's home page [NNg-ACC 14]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- pop-up windows, new windows
- Links

- **Test Description:**

This test is an implementation of the guideline # 14 on page 59 that says:

If you do open new browser windows, always provide a simple way to get back to the site's main homepage.

This test checks if the homepage can be reached by looking for links containing any words specified in parameter "Return to HOME label" (by default "home") in their label.

The test checks if the page opens new windows. If so, it checks whether opened windows contain links with any words specified in parameter "Return to HOME label" (by default "home") included in their labels.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

The page links/buttons open another page in a new window from which it appears there is no easy way to get back to the site's home page.

- **Test Parameters:**

- **Return to HOME** label with default value: `home` .

This parameter can be a bar separated list of words. It represents the **possible words that can be labels of a link or button**. These links or buttons are then assumed to lead to the site's home page. (Words are case-insensitive.)

More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

- **How to fix:**

Remember to put all the important navigation options in every page of the site. This will support the user navigation even in cases where the user is confused by new windows that are opened.

For example, placing a navigation bar that includes a link to the homepage at the top of the page provides the means to the user to keep navigating even in a newly created window.

Alternatively, you can place the link to the homepage behind the company logo on the top left-hand side of the page so that screen reader users will hear it as the first item on a page.

Finally, for a pop-up window, you can place a "Close Window" button on its top-left hand side so that users quickly realize it's there and can use it to close the new window.

- **Why to fix:**

We saw many cases where people using screen readers did not realize that several browser windows were open at once. If a site automatically opened another window, and the "Back" button failed them, people looked for a link to get to the homepage.

Pop-up windows were disorienting and often bewildering to people using screen readers and magnifiers, and often to people with motor skill disabilities as well. For people using screen readers, it can be a big problem if they click something, expect a certain page, and then hear something completely foreign like the contents of an unexpected pop-up window. The information they want, expect, and need is often right behind the pop-up window, but since the window focuses on the pop-up and that's what the screen reader reads, the user has no idea that what they want is right there.

One participant noted: "You press a link and you end up someplace way far from where you were and Back won't work. The only thing you can do is close the window." It is important to note that for users with screen readers or magnifiers, even a very small pop-up window eclipses the pages behind it.

We saw several examples where the users clicked a link and expected to go to a specific place, but an extra pop up window hindered their process. For example, a person who is blind might go to a large city's transit authority's website to figure out how to take public transportation. Many transportation sites offer a "Trip Planner" feature that allows users type in their starting street and their destination. They can also choose arrival and departure times. On one site we tested, when the user clicked the "Trip Planner" graphic link, the site took them to the Trip Planner form. However, there was a pop-up window in front of it that was selected. Although it's a small window, for people using screen readers

or magnifiers, it effectively eclipses the page behind it. In one instance, the screen reader read "Trip Planner" but the screen focus was still in the pop-up window. The user typed in her information, but the focus was not in a text box. She was lost and said, "I can't make heads or tails of this. I feel like such an idiot." On this same site, when a participant found the Trip Planner window after missing it several times because of the pop-up, she asked, "Why didn't I see this before? How did this appear? I didn't even see where it appeared as an option. I would have gone here first."

- **Learn More:**

- [tutorial on javascript windows](#)
- [tutorial on opening new windows with javascript](#)
- W3C on how to use [scripts](#)
- W3C/WAI techniques for making [accessible SCRIPTS](#) elements
- HTML 4.0 standard on the [SCRIPT](#) tag
- HTML 4.0 standard on [events](#)

13 cascading menus (16)

13.1 Avoid long cascading menus [NNg-ACC 16]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- cascading menus

- **Test Description:**

The test implements, in part, guideline # 16 described in page 64: "Avoid cascading menus". In particular this test flags all menus that contain too many entries (to handle the situations where you want to keep cascading menus but you want to limit their size).

The test checks all pages that contain a cascading menu (SELECT element) with missing SIZE attribute or with SIZE=1 (since they are rendered via a drop-down menu).

More specifically the test counts the number of options, or of option groups, within each SELECT element. If any of these numbers is greater than the value of the parameter "Max number of entries" (by default 10), then an issue is generated.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

Page contains cascading menus that contain more than N entries (where N is the value of the parameter "Max number of entries" (by default 10). A cascading menu is implemented through a SELECT element with size equal to 1. When the user selects it, the browser shows a list of options in a temporary area that disappears as soon as the mouse button is released.

- **Specific Issues:**

- **too many OPTIONS in menu:**
SELECT contains too many OPTIONS
- **too many OPTGROUPs and OPTIONS in menu:**
SELECT contains too many OPTGROUPs and OPTIONS elements

- **Test Parameters:**

- **Max number of entries** with default value: 10 .
This parameter can be any number greater than 0. It represents the **maximum number of menu entries** in a SELECT element.

- **How to fix:**

Consider restructuring the menus so that fewer choices are shown to the users. This can be achieved by splitting the set of choices into two or more separate menus.

- **Why to fix:**

Users of screen readers must listen to and remember all the entries of a menu before selecting the one they want. If the set of entries is larger than 7-10 items then it will become difficult for the user to do it without having to repeatedly listening the list.

Consider that this problem does not occur on cases where the user can easily memorize the items to be selected, like in the case of selecting a U.S. State from the list of their usual names.

Users of screen magnifiers often have a particularly difficult time using cascading menus. When the screen is zoomed in, often the cascading menu does not fit on the page. When the user scrolls the page to see the menu, their mouse moves from over the menu, and thus the menu closes and the user can never read it.

If you can, avoid using cascading menus altogether.

- **Learn More:**

- HTML 4.0 standard on [SELECT tag](#)
- HTML 4.0 standard on [forms in general](#)
- HTML 4.0 standard on the [FORM tag](#)
- W3C on [labeling form controls](#)

13.2 Avoid cascading menus [NNg-ACC 16]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- cascading menus

- **Test Description:**

The test implements guideline # 16 in page 64: "Avoid cascading menus". It flags all pages that contain a cascading menu (SELECT element) with missing SIZE attribute or with SIZE=1 (since they are rendered via a drop-down menu).

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

Page contains cascading menus. Avoid using them.

A cascading menu (or drop-down menu) is usually implemented via a SELECT tag with size = 1. In this way the browser shows only one entry of the menu, and when the user selects the menu, all the entries are shown in a drop-down temporary area.

- **How to fix:**

Avoid using cascading menus. Put the entries they contain into a list so that they are always visible. An easy way to accomplish it is to increase the size of the SELECT element (i.e. add SIZE="N" to the SELECT element, where "N" is the number of choices available). In this way the browser will display a scrollable list of N items.

- **Why to fix:**

People using screen magnifiers and people with motor skill challenges had a difficult time using cascading menus. These implementations rely on users being able to drag and hold the mouse while clicking with precision. Also, for screen magnifier users, the same issues that arose in our pop-up tests occurred with cascading menus. In some cases, the screen was very jumpy and hard to follow. In other cases, important items fell off the screen and were not visible. Flat menus are easier to use.

- **Learn More:**

- HTML 4.0 standard on [SELECT tag](#)
- HTML 4.0 standard on the [FORM tag](#)
- HTML 4.0 standard on [INPUT tag](#)
- HTML 4.0 standard on [possible types of INPUTs](#)
- W3C on [labeling form controls](#)
- HTML 4.0 standard on the [LABEL tag](#)
- about [Fitt's law](#)

13.3 Avoid pull-down menus [NNg-ACC 16]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- Manual
- NNg - beyond ALT text
- cascading menus
- links and buttons

- **Test Description:**

The test implements guideline # 16 in page 64: "Avoid cascading menus". It flags all pages that may contain a pull-down menu activated by a rollover.

In particular the test looks if the script activated by the MOUSEOVER event changes the visibility, xPos or yPos stylesheet properties of some part of the document.

The test is manual since LIFT cannot simulate the effect of a rollover.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

Page contains pull-down menus. Avoid using them.

When selecting a pull-down menu, the list of choices appears in a temporary area of the screen. But differently from the "drop-down" menus, this list is visible only while the mouse button is pressed. Releasing the mouse button will remove the temporary area.

A pull-down menus is usually implemented by a rollover, setting to "visible" the visibility property of an object, or changing the xPos or yPos stylesheet properties. This is achieved by a script inside a ONMOUSEOVER attribute of a link (which is usually the main menu item).

- **How to fix:**

Avoid using pull-down menus (and rollovers in general).

A better choice is to use drop-down menus (those with a small down arrow to their right: when you click them the list of options stays on the screen). They can be easily implemented in HTML via the SELECT tag.

Even better, consider restructuring the links that are displayed in the menu. Move some of them into more internal sections of the site, and possibly split them into groups and create a new page for each group.

- **Why to fix:**

People using screen magnifiers and people with motor skill challenges had a difficult time using cascading menus. These implementations rely on users being able to drag and hold the mouse while clicking with precision. This was extremely difficult and frustrating for users who had physical disabilities. Also, for screen magnifier users, the same issues that arose in our pop-up tests occurred with cascading menus. In some cases, the screen was very jumpy and hard to follow. In other cases, important items fell off the screen and were not visible. When they moved their mouse to show more of the zoomed in page, they needed to move off the menu, thus closing it. Sometimes they could never read the menu.

- **Learn More:**

- HTML 4.0 standard on [SELECT tag](#)
- W3C on how to use [scripts](#)

- HTML 4.0 standard on the [SCRIPT](#) tag
- HTML 4.0 standard on [events](#)
- about [Fitt's law](#)

14 links and buttons (17)

14.1 Avoid too many outgoing links [NNg-ACC 17]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- links and buttons
- Links

- **Test Description:**

This test implements guideline 17 (on page 65) that says: "Limit the number of links on a page".

The test flags pages that have more than a preset number of outgoing links. The number is stored in the parameter called "Max number of links" and is by default set to 20.

The test counts the number of links of type A or AREA that point to pages different than the current one.

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

Page contains too many links. Standalone links, those embedded in text, those included in navbars and those in imagemaps exceed a threshold set in the parameter called "Max. number of links", by default set to 20.

We recommend that each page have 20 links or less.

- **Test Parameters:**

- **Max number of links** with default value: 20 .
This parameter can be any number greater than 0. It represents the **maximum number of links** that can be contained in a page.

- **How to fix:**

Consider restructuring the links that are displayed in the page. Move some of them into more internal sections of the site, and possibly split them into groups and create a new page for each group.

Consider also if all the links are really needed. Are there redundant links that point to slightly different pages?

- **Why to fix:**

Using a screen reader and listening to many links on a page, or using a Braille reader and reading many links on a page is fatiguing. It is very difficult to remember the links and to keep them straight. Even 20 links is a lot for a user to listen through and remember enough of to make the right choice to advance to the next page.

Listening to a screen reader is like looking through the links on a page. While some users speed-up screen readers to read many words per minute, they still can only hear words one by one. Even manipulating the order in which the links are read does not change this fact. They cannot simply glance at things.

Some users listen to the whole page and don't click until they think the page has been completely read. One user said, "When it gets to stuff like privacy policy, that is usually a safe bet to assume that is the end of the page." This same user listened to 40 links before he realized he was on the wrong page. Other users click the first link they think might be right. The fewer links a page has, the more likely it is that people using screen readers will wait for the right one to be read.

One user with cerebral palsy used the arrow keys on his keyboard to move and scroll instead of using a mouse. Scrolling with the arrow keys made scrolling tedious and tiring. Pages that were very large or had many links exhausted him. It is also difficult for users with low vision to scroll to see many links. One user commented, "One site had 162 links, and that's a lot of reading in large print."

Users do expect some links on pages, however. Don't go too far the other way and have only one link on a page. One user encountered this and said: "If there is only one link you know it's really weird, or you bypassed it [the homepage]."

- **Learn More:**

- HTML 4.0 standard on [links in general](#) ;
- W3C/WAI on [grouping and bypassing links](#);
- HTML 4.0 standard on [image maps](#);
- a short tutorial on [accessible imagemaps](#);
- a tutorial on [accessible navigation](#)

15 small links (18)

15.1 Avoid too short text links or button labels [NNg-ACC 18]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- Links
- links and buttons

- **Test Description:**

This test implements guideline # 18 on page 65: "Avoid very small buttons and tiny text links".

This test flags A links or INPUT/type=button, INPUT/type=submit or BUTTON that have text labels with less than given # of chars (6 is by default the minimum allowed). It skips link images, banners and thumbnails. (These are dealt with by another test.) Hidden links are never considered.

This test does not consider font size. (Another test does it.)

The minimum number of characters that should appear as labels for a link is stored in the parameter called "Min. link length (chars)".

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

Page contains textual links or buttons that have labels that are too small (i.e. shorter than preset number of non blank characters). The preset value is stored in the parameter called "Min. link length (chars)". By default it is 6.

- **Specific Issues:**

- **link label too short:** Link has a label (that is, the textual content of the A element) that is too short.
- **BUTTON label too short:** BUTTON is too short.
- **INPUT label too short:** INPUT is too short.

- **Test Parameters:**

- **Min. link length (chars)** with default value: 6 .

This parameter can be any number greater than 0. It represents the **minimum length** (in characters except spaces) of the label of a textual A link (i.e. its content) or of a textual button.

- **How to fix:**

The easiest way to fix this problem is to make longer the link (or button) label. Think of additional words that you might use (like "Buy now" instead of "Buy") or add spaces so that the clickable area becomes larger.

If you use the A tag, put one or more before and after the currently used words in the content of the A element.

If you use BUTTON or INPUT, put one or more spaces before and after the words in the VALUE attribute of the element.

You can use the Fix Wizard to do this automatically for you.

- **Why to fix:**

For people with motor skill issues or low vision, aiming for a specific target can be difficult. It is frustrating for people when they accidentally miss the target, or worse, hit another target that is very close. It's difficult and can be tiring for people with motor skill challenges to hit small targets. In our studies, users with cerebral palsy using a touchpad or trackball took several tries to get the mouse to hit the small picture.

People with low vision experienced difficulty reading small links, created with a style sheet that specified 8-point type. In some cases the links were difficult to find because of their size and position.

- **Learn More:**

- HTML 4.0 standard on [links in general](#)
- HTML 4.0 standard on [BUTTON tag](#)
- HTML 4.0 standard on [INPUT tag](#)
- about [Fitt's law](#)
- CSS specification on [font size property](#)
- Alertbox on [why using and how to use CSS](#)

15.2 Avoid tiny links [NNg-ACC 18]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- Links
- links and buttons

- **Test Description:**

This test implements guideline # 18 on page 65: "Avoid very small buttons and tiny text links".

This test flags A links that have text labels with a font size that is too small. To determine font size, the test looks if the link specifies a CSS style rule with the FONT-SIZE property set to a value that is smaller than a predefined threshold.

The predefined thresholds for text size are stored in the parameter called "Too small font-size (symbols)", whose default value is: small|x-small|xx-small, and the parameter "Min. font-size (pixels)", whose default is 11.

If the link has a font-size corresponding to one of the words specified in the parameter "Too small font-size (symbols)", then the test flags the page with an issue.

Similarly, if the link has a font-size specified in points (pt) or pixels (px) whose value is smaller than the one stored in the parameter "Min. font-size (pixels)", then the test flags the page with an issue.

The test looks into in-line style rules, STYLE elements included in the page and external CSS files. If the problem lies with CSS rules specified in external files, the test creates an issue for each rule in the CSS that specifies a font-size that is too small for A links.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

Page contains textual links that appear to have a font size that is too small. In particular their font size matches the ones specified in the parameter called "Too small font-size (symbols)" (by default set to small|x-small|xx-small) or the font size is smaller than the value stored in "Min. font-size (pixels)" (by default set to 11).

- **Specific Issues:**

- **inline style FONT-SIZE is too small:** The inline CSS style specification contains a FONT-SIZE that is too small.
- **STYLE element with too small FONT-SIZE:** The page contains a STYLE element with a FONT-SIZE property for A links set to a value that is too small. (Note: there is an issue like this one for each FONT-SIZE that is too small.)
- **external CSS file with too small FONT-SIZE:** The page uses an external CSS file with a FONT-SIZE property for A links set to a value that is too small. (Note: there is an issue like this one for each FONT-SIZE that is too small.)

- **Test Parameters:**

- **Min. font-size (pixels)** with default value: 11 .
This parameter has to be a number. It represents the minimum size (in pixels) of the CSS property "font-size" for text and links inside the document.

- **Too small font-size (symbols)** with default value: `small|x-small|xx-small` .

This parameter can be a list of bar-separated words. Each word is a possible value of the CSS property "font-size". (Words are case insensitive).

More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

- **How to fix:**

The easiest way to fix this problem is to enlarge the font size of the text of the link. Change the SIZE attribute of the FONT tag or change the CSS property that applies to the link and enlarge the font size there.

Consider that the W3C discourages webdesigners to use the FONT tag in favor of a CSS style rule.

If you don't want to create an external style sheet, what you can do is to add the following code within the HEAD tag of your document in order to display all links using the default text size of the browser used by the visitor:

```
<STYLE>
  A {font-size: normal}
</STYLE>
```

You can use the Fix Wizard to add or change the font-size property of the currently selected link.

- **Why to fix:**

It's difficult and tiring for people with motor skill challenges (and some senior citizens) to hit small targets. One user with cerebral palsy was using a touchpad and took several tries to get the mouse to hit the small picture.

In several cases links were small because the style sheet specified 8-point type, making it difficult to read for people with low vision.

In other cases the links were difficult to find because of their size and position.

For people with motor skill issues or low vision, aiming for a specific target can be difficult. It is troubling and frustrating for people when they accidentally miss the target, or worse, hit another target that is very close.

- **Learn More:**

- Alertbox on [why using and how to use CSS](#)
- [CSS1 official specifications](#)
- HTML 4.0 standard on [adding styles to documents](#)
- HTML 4.0 standard on [the STYLE element](#)

- HTML 4.0 standard on [the style attribute](#)
- what the W3C says about [browsers default styles](#)
- HTML 4.0 standard on [links in general](#)
- a [CSS beginners guide](#)
- [what to do and what to avoid](#) when using CSS
- about [Fitt's law](#)

15.3 Avoid too small buttons [NNg-ACC 18]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- Links
- links and buttons

- **Test Description:**

This test implements guideline # 18 on page 65: "Avoid very small buttons and tiny text links".

This test flags A links containing images as labels that are narrower than 25 pixels or shorter than Min. link width (pixels) pixels (these are default values).

The minimum width and height of images that should appear as labels for a link are stored in the parameters called "Min. link width (pixels)" and "Min. link height (pixels)".

The test checks all the images within A links (except for spacers) and form buttons.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

Page contains the following images embedded within links that are too small (i.e. narrower than preset number of pixels **or** shorter than preset number of pixels).

The minimum width and height of images that should appear as labels for a link are stored in the parameters called "Min. link width (pixels)" and "Min. link height (pixels)". Their default values are 25 and 25.

- **Test Parameters:**

- **Min. link width (pixels)** with default value: 25 .

This parameter can be any number greater than 0. It represents the **minimum width** of the image that is the label of an A link (i.e. its content).

- **Min. link height (pixels)** with default value: 25 .

This parameter can be any number greater than 0. It represents the **minimum width** of the image that is the label of an A link (i.e. its content).

- **How to fix:**

Make buttons larger.

If you can, increase the width and height of the buttons so that they are greater than 25 pixels. If you need, choose new (and larger) images for the buttons.

- **Why to fix:**

It can be difficult and tiring for people with motor skill challenges to hit small targets. In our study, users with cerebral palsy using a touchpad or trackball took several tries to get the mouse to hit the small picture.

In some cases links were small because the style sheet specified 8-point type, making it difficult to read for people with low vision.

In other cases the links were difficult to find because of their size and position.

For people with low vision or motor skill issues, reading or aiming for a specific target can be difficult. It is troubling and frustrating for people when they accidentally miss the target, or worse, hit another target that is very close.

- **Learn More:**

- HTML 4.0 standard on [links in general](#)
- HTML 4.0 standard on the [IMG tag](#)
- HTML 4.0 standard on [INPUT tag](#)
- about [Fitt's law](#)

15.4 Avoid too small hotspots [NNg-ACC 18]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- links and buttons
- Links
- Imagemaps

- **Test Description:**

This test flags AREA elements with rectangular or circular shapes that have a size smaller than X or Y pixels (for circles they have to have a diameter smaller than X or Y).

X and Y are the values of the parameters called "Min. link width (pixels)" and "Min. link height (pixels)". Their default values are 25 and 25.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

Page contains hotspots (aka imagemaps) that are too small (i.e. narrower than X pixels or shorter than Y pixels).

X and Y are the values of the parameters called "Min. link width (pixels)" and "Min. link height (pixels)". Their default values are 25 and 25.

- **Test Parameters:**

- **Min. link width (pixels)** with default value: 25 .

This parameter can be any number greater than 0. It represents the **minimum width** of the image that is the label of an A link (i.e. its content).

- **Min. link height (pixels)** with default value: 25 .

This parameter can be any number greater than 0. It represents the **minimum width** of the image that is the label of an A link (i.e. its content).

- **How to fix:**

To fix this problem you have to make larger each region of the image map. In some cases you can simply extend each region (but remember to keep them separated enough). In other cases you may need to change the underlying image and choose a bigger one.

- **Why to fix:**

It can be difficult and tiring for people with motor skill challenges to hit small targets. In our studies, users with cerebral palsy using a touchpad or trackball took several tries to get the mouse to hit the small picture.

In some cases links were small because the style sheet specified 8-point type, making it difficult to read for people with low vision.

In other cases the links were difficult to find because of their size and position.

For people with motor skill issues or low vision, aiming for a specific target can be difficult. It is troubling and frustrating for people when they accidentally miss the target, or worse, hit another target that is very close.

- **Learn More:**

- HTML 4.0 standard on [AREA tag](#)

- HTML 4.0 standard on [image maps](#)
- HTML 4.0 standard on [possible shapes of client-side image maps](#)
- a short tutorial on [accessible imagemaps](#)
- about [Fitt's law](#)

16 separate links (19)

16.1 Leave space between textual links [NNg-ACC 19]

- **Test Type:** Automatic

- **Test Categories:**

- links and buttons
- NNg - beyond ALT text
- ALL TESTS

- **Test Description:**

The test implements guideline # 19 in page 67 that says: "Leave space between links and buttons". It checks that textual links are spaced by at least N characters or M pixels.

These thresholds are stored in the parameters "Min. distance (chars)" and "Min. horiz. distance (pixels)", respectively. Their default values are 3 characters and 25 pixels.

If pairs of links are separated by a mixture of text and images (for example spacers), then LIFT is unable to tell the actual distance between them. In this case the test does not generate issues.

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

Page contains textual links that are too close each other horizontally (not enough space between them). Links should be spaced by at least 3 characters or 25 pixels (default values of parameters called "Min. distance (chars)" and "Min. horiz. distance (pixels)").

- **Specific Issues:**

- **no horizontal space between links:** There is no space between two links that are horizontally aligned.
- **not enough text between textual links that are horizontally aligned:** There is not enough text between two textual links that are horizontally aligned.
- **not enough pixels between textual links that are horizontally aligned:** There are not enough pixels between two textual links that are horizontally aligned.

- **Test Parameters:**

- **Min. distance (chars)** with default value: 3 .

This parameter can be any number greater than 0. It represents the **minimum number of text characters** that have to be placed between adjacent text links in order to consider the links sufficiently well separated.

- Images
- NNg - beyond ALT text
- ALL TESTS

- **Test Description:**

The test implements guideline # 19 in page 67 that says: "Leave space between links and buttons". It checks that buttons are spaced by at least N characters or M pixels horizontally and K pixels vertically.

These thresholds are stored in the parameters "Min. distance (chars)", "Min. horiz. distance (pixels)" and "Min. vert. distance (pixels)". Their default values are 3, 25 and 10.

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

Page contains buttons (links having images as labels) that are too close each other (not enough blank pixels or text between them), either horizontally or vertically.

Used thresholds are stored in the parameters "Min. distance (chars)", "Min. horiz. distance (pixels)" and "Min. vert. distance (pixels)".

- **Specific Issues:**

- **no horizontal space between buttons:** There is too little space between two buttons that are horizontally aligned.
- **not enough text between buttons that are horizontally aligned:** There is not enough text between two buttons that are horizontally aligned.
- **not enough pixels between buttons that are horizontally aligned:** There are not enough pixels between two buttons that are horizontally aligned.
- **no vertical space between buttons:** There is too little space between two buttons that are vertically aligned.
- **not enough text between buttons that are vertically aligned:** There is not enough text between two buttons that are vertically aligned.
- **not enough pixels between buttons that are vertically aligned:** There are not enough pixels between two buttons that are vertically aligned.

- **Test Parameters:**

- **Min. distance (chars)** with default value: 3 .

This parameter can be any number greater than 0. It represents the **minimum number of text characters** that have to be placed between adjacent text links in order to consider the links sufficiently well separated.

- **Min. horiz. distance (pixels)** with default value: 25 .
This parameter can be any number greater than 0. It represents, in pixels, the **minimum width** of images that have to be placed between horizontally adjacent links in order to consider the links sufficiently well separated.
- **Min. vert. distance (pixels)** with default value: 10 .
This parameter can be any number greater than 0. It represents, in pixels, the **minimum height** of images that have to be placed between vertically adjacent links in order to consider the links sufficiently well separated.

- **How to fix:**

LIFT is unable to detect under all conditions if the suspicious buttons are actually closer than the threshold (this often occurs when text is mixed with images). You need to check manually if selected buttons are too close each other.

Another thing that LIFT is unable to check is whether the buttons have a central focal area where the user presumably clicks and if this area is surrounded by a background that already separates pairs of buttons. You should check manually if this is the case. If so, you don't have to worry about this test.

There are many ways to keep apart pairs of links:

- if you use text to separate buttons, you can use ` ` to add invisible spaces in the text between pairs of links
- you can use `”style=’line-height: .02em”` within the `A` element to enlarge vertically the line in which the button is placed
- you can change the size of the images that are placed between pairs of links
- you can add spacers (transparent 1x1 images) between buttons (remember to set `ALT=”` for them)

- **Why to fix:**

For people with motor skill issues or low vision, space between buttons and links is extremely important.

Aiming for a specific target can be difficult, and it is troubling and frustrating for people when they accidentally miss the target, or worse, hit another target that is very close.

Sighted users also sometimes accidentally hit the wrong links when links appear too close together. Leaving blank space between links and buttons helps eliminate this problem.

In our studies, we saw several users with physical disabilities click the wrong link when links were too close together. For people with low vision, having white space between links can also help readability.

- **Learn More:**

- HTML 4.0 standard on [links in general](#)
- HTML 4.0 standard on [the style attribute](#)

- HTML 4.0 standard on [the STYLE element](#)
- HTML 4.0 standard on [adding styles to documents](#)
- CSS specification on [line height property](#)
- Alertbox on [why using and how to use CSS](#)
- about [Fitt's law](#)

16.3 Leave space between vertically aligned textual links [NNg-ACC 19]

- **Test Type:** Automatic
- **Test Categories:**
 - links and buttons
 - Links
 - NNg - beyond ALT text
 - ALL TESTS

- **Test Description:**

The test implements guideline # 19 in page 67 that says: "Leave space between links and buttons". It checks if textual links that are vertically aligned using BR tags are sufficiently separated. This test skips lines that specify an inline CSS property "line-height" with a value of 0.2em or larger.

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

Page contains text links that are too close together. They are aligned vertically and are separated by a BR tag only. (This test skips lines that specify an inline CSS property "line-height" with a value of 0.2em or larger.)

- **How to fix:**

Increase the vertical separation between links.

There are many ways to keep apart pairs of vertically aligned links:

- you can use "style='line-height: 0.2em'" within the A element to enlarge vertically the line in which the button is placed.
You can use the Fix Wizard to do this automatically for you.
- if you use a table to layout links, you can add empty rows (a TD cell containing) to separate the links
- you can change the size of the images that are placed between pairs of links
- you can add spacers (transparent 1x1 images) between links (remember to set ALT="" for them)

- **Why to fix:**

For people with physical disabilities that include motor skill issues or low vision, space between buttons and links is extremely important.

Aiming for a small target target is difficult, and can be very frustrating for people when they miss the target, or worse, accidentally hit another target that is very close.

Note that even sighted users also sometimes accidentally click the wrong links when links appear too close together. Leaving blank space between links and buttons helps eliminate this problem.

In our studies, we saw many users accidentally clicking the wrong link because it was situated very close to the link they were aiming for.

- **Learn More:**

- HTML 4.0 standard on [the style attribute](#)
- CSS specification on [line height property](#)
- about [Fitt's law](#)
- Alertbox on [why using and how to use CSS](#)
- [CSS1 official specifications](#)
- a [CSS beginners guide](#)
- [browsers compatibility chart](#)
- detailed [CSS browsers known bugs](#)
- [what to do and what to avoid](#) when using CSS

17 underline links (22)

17.1 Underline all links [NNg-ACC 22]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- Links
- links and buttons

- **Test Description:**

This test searches Cascade Style Sheets rules for A:link, A:visited, A:active with text-decoration: none. It looks into the inline style attribute, the STYLE of the page and in the style file if external.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

Page contains Cascade Style Sheets rules that disable link underlining.

Note: if the CSS rule that disables link underlining is stored in an external style file, then this issue will occur on all the pages that use that style sheet. We suggest that, after you fix this problem, you rerun LIFT to remove all the issues.

- **Specific Issues:**

- **local CSS disable link underlining:** CSS in STYLE element of this page includes text-decoration different than underline.
- **external CSS disable link underlining:** CSS in external file specified in LINK element includes text-decoration different than underline.

- **How to fix:**

Fixing this problem is really easy. Change the style specification for links (A elements) and either remove the property **text-decoration** (to restore the browser default) or set it to **underline**.

You can use the Fix Wizard to do this automatically for you.

Consider that it is important to have underlined links **before** the mouse gets over them. In this way users, when they scan the page, can easily locate links.

Underlining links only when the mouse passes over them makes them much less prominent. Only after the user has thought of passing the mouse over them, they will display the underline. But if the mouse never hovers them, then the user might not discover that there is a link.

- **Why to fix:**

This is a common Web design recommendation but is worth repeating in this accessibility context. (Note that underlining links is a browser function; designer shouldn't try to defeat it.) Underlines are especially important for users with low vision. When a page is zoomed in, it is difficult to discern some of the more subtle link indicators, like bold or blue text. Underlines are easier to see when the screen is magnified.

In some of our studies, users found it difficult to tell which items were banners or text, and which were links. They had a much easier time using sites that underline all links.

- **Learn More:**

- [Alertbox on why using and how to use CSS](#)
- [CSS1 official specifications](#)
- [CSS specification on text decoration property](#)
- [HTML 4.0 standard on adding styles to documents](#)
- [HTML 4.0 standard on the STYLE element](#)
- [what the W3C says about browsers default styles](#)
- [detailed CSS browsers known bugs](#)
- [browsers compatibility chart](#)

18 links and buttons (23)

18.1 Avoid embedding image buttons in text [NNg-ACC 23]

- **Test Type:** Automatic
- **Test Categories:**
 - ALL TESTS
 - NNg - beyond ALT text
 - Links
 - Images

- **Test Description:**

This test implements guideline # 23 on page 72: "Create links within text when it makes sense. Use additional buttons only when it's necessary".

The test checks if the page contains repeated link buttons (that is, A elements containing IMGs that is repeated more than once in a page) that are embedded within text.

More specifically, only images that are repeated two or more times in a page are considered by this test.

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

The page contains images used as buttons that are embedded within text.

- **How to fix:**

Replace the button with a textual link.

- **Why to fix:**

If it is simple to create an underlined link to take people to the next page, do this.

When screen magnifier users have zoomed in on a page, it is much easier to keep context when the link is actually the explanatory text instead of a button near the text. Some sites describe the page the link leads to, then provide a "go" or similar button. This is less elegant for both screen reader and screen magnifier users. (It can also cause problems for those users who turn off graphics.)

We saw many examples when users would have been better served with a simple text link instead of a button after the text. For example, on a financial website, text described some financial services, and was followed by a "Go" button which linked to descriptions of those services. First, the "Go" buttons were too far away from the actual descriptions they related to. They should have been closer. Also, the word "Go" did not descriptively relate to the feature it linked to- it is too generic. Most importantly, users tried to click in the text which was bolded, to get to the descriptions. As they were reading, it was more intuitive to click text of the actual description that looked like a link than it was to look for a

button after the fact. After a user with low vision finally saw the "Go" button, she said, "I didn't see the Go. Maybe they should have put the Go to the right instead of underneath." Another user who didn't see the Go button said, "There should be a hyperlink for that."

- **Learn More:**

- HTML 4.0 standard on the [IMG tag](#);
- W3C on [how to define the ALT attribute](#);

19 confirm what the page is (25)

19.1 Confirm what the page is: check its title [NNg-ACC 25]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- Title

- **Test Description:**

This test implements guideline # 25 on page 74 that says: "Immediately confirm what the page is once it has loaded".

The test checks if the page has more than one title, that the title is not empty, that it does not contain words contained in parameter "Invalid title content" (by default untitled).

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

Page title is not well defined.

- **Specific Issues:**

- **more than one title:**
TITLE tag is duplicated (different browsers will pick up different titles)
- **missing title:** There is no TITLE tag
- **empty title:**
TITLE has no content
- **title contains placeholders:**
TITLE contains only placeholder text

- **Test Parameters:**

- **Invalid title content** with default value: `untitled` .

This parameter can be a bar separated list of words. It represents the **possible words that can be contained in the page title** to qualify the title as being not significant. (Words are case-insensitive.)

More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

- **How to fix:**

To define the title of the page include the following HTML code just below the `<HTML>` tag at the beginning of your document:

```
<HEAD>
  <TITLE>
    Buy the best strawberries fields in the world
  </TITLE>
</HEAD>
```

With Dreamweaver you can do it in a simpler way: just fill-in the title box on top of the Editor window.

You can also use the Fix Wizard to edit the title of the page and to remove duplicated titles.

Consider that titles, to be effective, have to satisfy the following conditions:

- a title should be **brief** and **informative**
- should not contain HTML tags
- should be defined only once in a document.

- **Why to fix:**

People using screen readers or Braille devices rely on the page title to indicate that they are on the page they expect to be on. And since URL's can be complex, it is always important to briefly state the purpose of the page.

The TITLE tag defines the document's title for:

- browser windows
- speaking browsers
- bookmark lists
- history lists
- search engine results lists

A poor, or missing, TITLE would make the page much more difficult to locate and understand in any situation where the user is required to identify a window, figure out the context, or select an item from a bookmark list or search results.

To be effective, titles should be **brief** and **informative** (i.e. adequately describe the page).

... a page title needs to be a pearl of clarity.

- **Learn More:**

- HTML 4.0 standard on the [TITLE tag](#)
- Alertbox on [page titles](#)
- Alertbox on [how to write](#) effective page titles.

20 homepage logo (26)

20.1 Do not associate the word 'homepage' with your company logo [NNG-ACC 26]

- **Test Type:** Automatic
- **Test Categories:**
 - ALL TESTS
 - NNG - beyond ALT text
 - page organization
 - Images

- **Test Description:**

This test implements guideline # 26 on page 74: "Do not associate the word homepage with your company logo if you plan to reuse the same graphic on all pages".

The test checks if the page contains the company logo (defined in the parameter "Logo file", by default set to "logo.gif"). If so, it checks if it has an ALT. If no ALT is defined, or if it is not valid, an issue is raised.

If the ALT is defined, and if the logo is contained within a link (i.e. the image is clickable), unless the ALT contains the words specified in the parameter "Words in logo ALT", an issue is generated. The default value of the parameter is "link to|go to|open|view".

This test has **priority 2**: users were very frustrated when this guideline was violated.

NOTE: this test is redundant with another one that checks for validity of the ALT text of any image, including logos. This redundancy is intentional: in this way with a single issue all the problems of an ALT are highlighted.

- **Issue Description:**

The page contains the logo image (specified in the parameter "Logo file") with no properly defined ALT.

- **Specific Issues:**

- **ALT not defined:** No ALT defined for logo image.
- **ALT is empty:** ALT of logo image is the empty string "".
- **ALT is blank:** ALT of logo image is the blank string " ".
- **ALT with HTML tags:** ALT of logo image contains HTML tags.
- **ALT too long:** ALT of logo image is too long.
- **ALT with size:** ALT of logo image describes only the size of the image file.
- **ALT with filename:** ALT of logo image is the filename of the image.

- **ALT with placeholder text:** ALT of logo image appears to contain only placeholder text.
- **required text is missing:** ALT of logo image does not contain the required words specified in parameter "Words in logo ALT".

- **Test Parameters:**

- **Logo file** with default value: `logo.gif` .
This parameter has to be a filename. It represents part of the pathname of the file containing the company logo. It is used to identify the logo in a page: any image whose SRC attribute contains this filename is supposed to be the logo.
- **Words in logo ALT** with default value: `link to|go to|open|view` .
This parameter can be a bar separated list of words (regular expressions). It represents the mandatory part of the ALT associated to the logo image, if any. The logo file is identified by its name, stored in the parameter "Logo file".
More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

- **How to fix:**

When naming the logo graphic, be clear in the ALT text and the description as to : 1) where the user is, and 2) where clicking the logo will bring them.

- **Why to fix:**

Screen reader users, during our studies, were frequently unsure of the site they landed on. While a sighted person might see a company logo, a screen reader user often just heard a cryptic URL being read. It's important that sites use both TITLE tags to name pages and ALT text on the logo, or provide the name of the site in text very near the top of the page. Often sites use the company logo to link to the homepage, which is a good practice. This is very confusing though when the user is already on the homepage and the user thinks they need to click the graphic to get to the homepage. Similarly, when a logo is called 'homepage', meaning "click this to get to the homepage", often users thought it meant they were already on the homepage.

Some screen readers are set to always read the window title (TITLE tag) and others start with the first link or the first text, depending on the mode they are in when the page is loaded. A participant using a screen reader hit the Major League Baseball homepage and said, "I like this because it tells me the official site of Major League Baseball right away."

Good page organization means different things for sighted users, screen reader users, and screen magnifier users. To help demonstrate this, we deconstruct the White House's homepage, www.whitehouse.gov.

One of the very first things a screen reader reads, is "Welcome to the White House". While writing this on a homepage seems outdated, for screen reader users, it is helpful to hear the page title. It confirms to the user, right away, that they have hit the right site. This

particular example is noteworthy because the same address with a .com extension instead of a .gov extension is an adult site.

The White House home page does a nice job of organizing the order of what is read. Even though these links are invisible to the eye, one of the first options a screen reader will read is "skip to content". This lets you quickly skip the navigation area and go straight to the page content.

And, even though this option is visibly available at the very end of the page, "text only" is also one of the first links a screen reader will read. Finally, "search" is also among the first audible links. This lets users know that search is available and that if they want to, they can easily select it and skip listening to all the text. This handy trick is easily accomplished by using either small, visible graphics or transparent graphics at the top of the page that link to the various destinations. These invisible links are available to both screen reader users and browsers that provide link lists.

- **Learn More:**

- W3C on [how to define the ALT attribute](#);
- HTML 4.0 standard on the [IMG tag](#);
- Alertbox on [how to write effective page titles](#);
- HTML 4.0 standard on the [TITLE tag](#);
- Alertbox on [ALT descriptions](#);
- HTML 4.0 standard on [links in general](#) ;
- another tutorial on [alternative content for graphics](#) rich of examples;
- a tutorial on the [use of ALT text in IMGs](#);

21 skip links (31)

21.1 Use 'skip links' so users can skip links or navigational elements [NNg-ACC 31]

- **Test Type:** Automatic
- **Test Categories:**
 - ALL TESTS
 - NNg - beyond ALT text
 - page organization
 - Links

- **Test Description:**

This test implements guideline # 31 on page 78: "Carefully consider using 'skip links' so users can skip links or navigational elements".

The test scans all the navigation bars within the page. If some of them does not follow a **hidden link** that enables to skip the navigation bar, the test prompts the user to verify if such a link is needed.

This test assumes that the **hidden link** mechanism (see below) is used to implement this requirement. The test also assumes that the hidden link is located to the left and close to the navigation bar.

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

The page contains a navigation bar that does not follow a **hidden link**. A hidden link (in the context of this test) is a link that is normally not shown by the browser and is implemented like the following example:

```
<A href="#content">
  <IMG alt="jump over products navigation bar" src="spacer.gif"
    width="1" height="1">
</A>
```

and is used to jump over a navigation bar directly to a content section of the page.

NOTE: if there is some alternative mechanism for skipping the navigation bar that this test was not able to identify, you should not worry about this issue.

In addition, if this page is framed (i.e. it is the content of a FRAME) and frames have appropriate titles, then you **don't need** to implement what is suggested by this test, since a frame containing links is already a skippable group of links.

- **Specific Issues:**

- **missing hidden link:** Couldn't find hidden links used to skip navigation bar.
- **missing named anchor:** Hidden link doesn't point to a fragment of the document.

- **How to fix:**

The easiest way to satisfy this requirement is to place a normal textual link (labeled like "skip navigationals") that points to a named anchor in the same page just where the actual content begins.

A more effective way that does not affect the visual appearance of the page is to put, just before the navigation bar, a link pointing to the content of the page whose label is a transparent gif with an appropriate ALT defined. For example:

```
<A href="#content">
  <IMG alt="jump over products navigation bar" src="spacer.gif"
    width="1" height="1">
</A>
```

In this way, users of graphical browsers would not see that link. It can be used, however, by users of non-graphical browsers.

The ALT description should describe what is being skipped, to inform the user about its existence and purpose.

You can also place the **hidden link** at the beginning of the page if you want to provide a single link for skipping one or more groups of links.

Another way to implement the hidden link is to set the color of the text to be same as the background color. The link would not be perceived when colors are shown. Consider however that color of the text is usually not under your control, but the user can in general change it with the browser.

And yet another way to implement this requirement is to use frames. In fact frames can be used to group links and, if they contain informative titles, then they can support navigation very well.

- **Why to fix:**

Good page organization means different things for sighted users, screen reader users, and screen magnifier users. To help demonstrate this, we deconstruct the White House's homepage, www.whitehouse.gov.

One of the very first things a screen reader reads, is "Welcome to the White House". While writing this on a homepage seems outdated, for screen reader users, it is helpful to hear the page title. It confirms to the user, right away, that they have hit the right site. This particular example is noteworthy because the same address with a .com extension instead of a .gov extension is an adult site.

The White House home page does a nice job of organizing the order of what is read. Even though these links are invisible to the eye, one of the first options a screen reader will read is "skip to content". This lets you quickly skip the navigation area and go straight to the page content.

And, even though this option is visibly available at the very end of the page, "text only" is also one of the first links a screen reader will read. Finally, "search" is also among the first audible links. This lets users know that search is available and that if they want to, they can easily select it and skip listening to all the text. This handy trick is easily accomplished by using either small, visible graphics or transparent graphics at the top of the page that link to the various destinations. These invisible links are available to both screen reader users and browsers that provide link lists.

When sites have a top navigation bar on all pages, it can be time-consuming for screen reader users to have to listen to it on every page on a site. They are after content, and can become frustrated when they must first wade through all the links at the top of a page. (This is also true when they are reading search results.) "Skip links" can help users avoid tedious repetitions of every single top and side navigation on every page.

- **Learn More:**

- HTML 4.0 standard on [links in general](#)
- HTML 4.0 standard on the [anchor names](#);
- W3C/WAI on [grouping and bypassing links](#)
- a [detailed discussion](#) on linearizing tables and its effects on navigation;

22 limit info required by forms (35)

22.1 Limit amount of information required by forms [NNg-ACC 35]

- **Test Type:** Automatic
- **Test Categories:**
 - ALL TESTS
 - NNg - beyond ALT text
 - Forms
 - forms and fields

- **Test Description:**

This test implements guideline # 35 on page 82: "Limit the amount of information that forms require; collect only the minimum needed".

This test flags forms that contain too many text input fields and/or too many controls in general.

The test counts the text input fields (i.e. INPUT elements with TYPE=text and TEXTAREA elements that are not hidden, disabled nor read only) and compares this number with the value of the parameter called "Max num. of text fields", whose default value is 10.

The test also counts the total number of controls that are active within a form (i.e. INPUT, SELECT, BUTTON, TEXTAREA elements that are not hidden, disabled nor read only) and compares this number with the value of the parameter called "Max num. of form controls", whose default value is 20.

This test has **priority 3**: user experienced little frustration when this guideline was violated.

- **Issue Description:**

The page contains a form that requires too much information: it has too many text fields and/or too many controls in general (selection boxes, radio/check-buttons, buttons, text fields).

- **Specific Issues:**

- **Form has too many text fields:** The form has a number of text fields that is greater than the maximum limit, stored in the parameter "Max num. of text fields".
- **Form has too many controls:** The form has a number of controls that is greater than the maximum limit, stored in the parameter "Max num. of form controls".

- **Test Parameters:**

- **Max num. of text fields** with default value: 10 .
This parameter represents the maximum number of active text fields that can be present in a form.

- **Max num. of form controls** with default value: 20 .

This parameter represents the maximum number of active text fields that can be present in a form.

- **How to fix:**

Please double check that all the information required by the form is actually needed. If some information is not needed at the current stage of the interaction, or if it has been previously asked, avoid requiring it in this form.

- **Why to fix:**

Some users with cerebral palsy had difficulty typing with precision, even when they were using a keyguard. This was especially true after testing for more than 30 minutes or so. They also had difficulty filling in very long forms. The JWA site presents an order form logically. However, the form had so many items that users had to scroll a lot to see them all. One participant using a screen magnifier could not find a way to scroll down to order a book. He had magnified the page so that the right hand scrollbar was not visible, and when he moved the mouse pointer to the scroll bar, the book image on the left disappeared.

- **Learn More:**

- HTML 4.0 standard on the [FORM tag](#);
- HTML 4.0 standard on the [LABEL tag](#);
- HTML 4.0 standard on [forms in general](#);
- W3C on [labeling form controls](#);
- a [tutorial on accessible forms](#)

23 label close to field (36)

23.1 Put text label very close to field [NNG-ACC 36]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- Manual
- NNG - beyond ALT text
- Forms
- forms and fields

- **Test Description:**

This test implements guideline # 36 on page 83: "Put text for field labels very close to the actual field".

This test flags forms that contain field labels that are not in the same table cell of the actual field. The test fires on when the form is being layed out using tables. It assumes that putting the control on one table cell and its label in another cell may lead to violations of this guideline.

The test asks the user to check if the field is very close to the corresponding text label.

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

The page contains a form with controls that may be too far from their labels. Remember to put text for field labels very close to the actual field.

- **How to fix:**

Check the distance between the form controls and their labels. Try to increase font size of the browser and see if the distance increases and how much. Try to use a screen magnifier to determine under which circumstances the text label is not seen together with the control it refers to.

Avoid using a visual layout that separates the form controls from its labels. Use the same background color and use visual clues to group form controls with their labels.

Remember that even if you used the LABEL element to mark the text that is intended to be the label for some form control, screen magnifiers do not take advantage of it.

- **Why to fix:**

When forms are magnified, parts of the page disappear. The farther a field's label is from its entry field, the more likely it is that magnified screen users will not see the label that accompanies the field entry box.

Some users are aware of this and know they must scroll to see the label. Others look to the text that is visible and assume that the text closest to the open field belongs with that field.

Also, when people increase font size through their browser, the designer's intended positioning of items can change.

- **Learn More:**

- HTML 4.0 standard on the [FORM tag](#);
- HTML 4.0 standard on the [LABEL tag](#);
- HTML 4.0 standard on [forms in general](#);
- W3C on [labeling form controls](#);

24 color coding (37)

24.1 Avoid color coding of form errors [NNg-ACC 37]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- Manual
- NNg - beyond ALT text
- Forms
- forms and fields

- **Test Description:**

This test implements guideline # 37 on page 83: "Do not use only red or yellow highlighting to indicate form errors".

This test flags forms with one or more text input fields that may generate error messages that are color coded.

The test asks the user to check if the form uses colors alone to indicate errors. In particular the test does not examine color properties in the HTML of the page since error messages are usually supplied by the webserver (and are unavailable to LIFT).

The test is manual since in general it is difficult to test if a form generates errors and how errors are coded, since this depends on how the server processing the form works.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

The page contains a form that may generate errors that are described using colors alone (typically red or yellow highlighting to indicate missing or wrong input).

- **How to fix:**

Please check that **none** of the possible errors generated by filling-in the form are color coded.

If they are color coded, then consider the following remedies.

A better implementation for helping users complete their form is to show them only the part of the form that has errors. Rather than making them go through the entire form again (and wonder if fields they've completed correctly might be wrong) you can show them only fields they need to fix or fill in. If the form has many errors, let them whittle their way through it until everything is correct.

Alternatively, clearly mark the fields that are wrong using text (for example, by using the word "ERROR" in the label of the wrong fields).

- **Why to fix:**

While it is essential to tell users the exact location of any form errors, red text is not the best indicator. Screen reader users and Braille users have no way of knowing which text is red.

On a magnified screen, red text is somewhat difficult to read. Also, screen magnifier users often invert colors to see things better. When they do this, the errors are no longer red. When looking at an error page, one participant with low vision said, "I don't see any highlighted fields."

- **Learn More:**

- HTML 4.0 standard on [color names and color usage](#);
- specific [techniques](#) suggested by WAI to handle colors in accessible pages;
- How color blind view the world: [a short tutorial](#)
- a tutorial (by LightHouse.org) for achieving effective [color contrast](#)
- HTML 4.0 standard on the [FORM tag](#);
- HTML 4.0 standard on the [LABEL tag](#);
- W3C on [labeling form controls](#);

25 required fields (38)

25.1 Do not rely only on asterisk [NNg-ACC 38]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- Manual
- NNg - beyond ALT text
- Forms
- forms and fields

- **Test Description:**

This test implements guideline # 38 on page 83: "Do not rely only on asterisk (*) to indicate required fields".

This test flags forms that contain asterisks (*).

The test asks the user to check if the asterisk is the only way that is used to indicate that a field or control is mandatory.

The test is manual since LIFT is unable to tell the meaning of the asterisk.

This test has **priority 3**: user experienced little frustration when this guideline was violated.

- **Issue Description:**

The page contains a form that may use asterisks (*) to indicate that fields are required.

- **How to fix:**

Please check if the form uses asterisks to indicate required fields.

If so, you can still use asterisks to indicate required fields, as many sighted users know this convention. However, we recommend that you also provide another indicator.

Bolded text might work for some users.

Better would be to only request required information.

Other than that, consider organizing the page so that all required fields appear at the top of the form, and all other fields appear toward the bottom. Optional fields could also appear in a separate section, at the bottom of the form.

- **Why to fix:**

A screen magnifier reads an asterisk as "star". Some screen reader users have learned that the word "star" means required field, but several users in our study did not know this. A few even said that they didn't know why they sometimes heard the word "star", and thought it might be a bug in their screen reader.

- **Learn More:**

- HTML 4.0 standard on the [FORM tag](#);
- HTML 4.0 standard on the [LABEL tag](#);
- W3C on [labeling form controls](#);
- HTML 4.0 standard on [color names and color usage](#);
- specific [techniques](#) suggested by WAI to handle colors in accessible pages;
- a tutorial (by LightHouse.org) for achieving effective [color contrast](#);
- How color blind view the world: [a short tutorial](#);

26 logical tab order (39)

26.1 Make sure tab order is logical [NNg-ACC 39]

- **Test Type:** Manual
- **Test Categories:**
 - ALL TESTS
 - Manual
 - NNg - beyond ALT text
 - Forms
 - forms and fields

- **Test Description:**

This test implements guideline # 39 on page 85: "Make sure the tab order is logical".

This test flags forms that contain at least two controls (fields, buttons, selection lists, radio buttons or checkboxes).

The test asks the user to check if the tab order is logical.

The test is manual since LIFT is unable to tell what is a logical tab order.

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

The page contains a form that may lay out fields and other controls with a tabbing order that is not logical.

- **How to fix:**

Please check if the form uses a tab order that is not logical. Try moving through the form controls by using only the TAB key on your keyboard. Is the focus moving through them as you would expect? Is the visual layout of the fields followed when using the TAB key?

It may happen that the form fields are layed out on two columns and the intended order is top to bottom on first column and then the same on the second one. But the TAB order may turn out to be first field in column one, followed by first field in column 2, and then similarly for second field, and so on. (This problem is due to an incorrect **linearization** of the table used to lay out the form.)

To fix these problems you can group the fields of each column within a subtable. Use one subtable for each field column. In this way the whole content of a column will be scanned before moving the focus to the subsequent columns. (However this solution will increase the number of tables used in the page, which is not a good idea.)

An even better solution is to use the TABINDEX attribute, to be associated to each control in the form. Its value should be a number that specifies the order in which that control will receive the focus. Beware, however, that all the controls in the page (not only within

the form) are affected by the TABINDEX. Therefore also all the links, hotspots, buttons might be indirectly affected. If you define the TABINDEX for a form, do a thorough test of all the controls in the entire page.

- **Why to fix:**

Many people use the Tab key to move through fields when filling in a form. For some people, this is simply easier. For others, including most of our study's participants, using a mouse is impossible and TAB is their only option.

People listening to screen readers are especially dependent on the field order being logical because that is how the reader leads them through the form. We found that many sites already have good, logical tab order.

- **Learn More:**

- HTML 4.0 standard on the [FORM tag](#);
- HTML 4.0 standard on the [TABINDEX attribute](#);
- a [detailed discussion](#) on linearizing tables;
- HTML 4.0 standard on the [ACCESSKEY attribute](#) (to define keyboard shortcuts).

27 match visual order (40)

27.1 Match tab order to visual layout [NNG-ACC 40]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- Manual
- NNG - beyond ALT text
- Forms
- forms and fields

- **Test Description:**

This test implements guideline # 40 on page 86: "Match the tab order to the visual layout when possible".

This test flags forms that contain at least two controls (fields, buttons, selection lists, radio buttons or checkboxes).

The test asks the user to check if the tab order matches the visual layout.

The test is manual since LIFT is unable to tell what is the visual layout of a general form.

This test has **priority 3**: user experienced little frustration when this guideline was violated.

- **Issue Description:**

The page contains a form that may lay out fields and other controls with a tab order that does not match the visual layout.

- **How to fix:**

Please check if the form uses a tab order that matches the visual layout. Try moving through the form controls by using only the TAB key on your keyboard. Is the focus moving through them as you would expect? Is the visual layout of the fields followed when using the TAB key?

It may happen that the form fields are layed out on two columns and the intended order is top to bottom on first column and then the same on the second one. But the TAB order may turn out to be first field in column one, followed by first field in column 2, and then similarly for second field, and so on. (This problem is due to an incorrect **linearization** of the table used to lay out the form.)

To fix these problems you can group the fields of each column within a subtable. Use one subtable for each field column. In this way the whole content of a column will be scanned before moving the focus to the subsequent columns.

An even better solution is to use the TABINDEX attribute, to be associated to each control in the form. Its value should be a number that specifies the order in which that control will receive the focus. Beware, however, that all the controls in the page (not only within the form) are affected by the TABINDEX. Therefore also all the links, hotspots, buttons might be indirectly affected. If you define the TABINDEX for a form, do a thorough test of all the controls in the entire page.

- **Why to fix:**

In one of our studies, on a website, the tab order was logical, but did not match the visual design. So, while this worked for screen reader users, sighted users and those with low vision were thrown off.

The tab order went like this:

```
First name
  Middle initial
  Last name
  Address 1
  Address 2
  City
  State
  Zip/postal code
  E-mail
  Please reenter e-mail
  Phone
```

The layout went like this, so "phone" came before "please reenter email":

```
First name Middle initial Last name
Address 1 Address 2
City State Zip/postal code
E-mail Phone
Please reenter e-mail
```

The tab order was logical (email, re-enter email, phone), but the layout contradicted the tab order. For most fields, the tab order went left to right. In the visual layout, the reenter e-mail field appeared underneath the e-mail field. When the user tabbed through the fields, they were thrown off because, while logical, the fields broke with the design and went down, then up and right.

- **Learn More:**

- HTML 4.0 standard on the [FORM tag](#);
- HTML 4.0 standard on the [TABINDEX attribute](#);
- a [detailed discussion](#) on linearizing tables;

28 stack fields (41)

28.1 Stack fields in a vertical column [NNg-ACC 41]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- Forms
- forms and fields

- **Test Description:**

This test implements guideline # 41 on page 86: "Stack fields in a vertical column".

The test checks if the page contains a form with controls arranged in more than one column of a layout table.

More specifically, it checks if some control of the form (except for buttons) is positioned in more than one table column.

This test has **priority 3**: user experienced little frustration when this guideline was violated.

- **Issue Description:**

The page contains a form with controls arranged in more than one column.

- **How to fix:**

Consider restructuring the form putting all text fields, menus, radio buttons and checkboxes on the same column.

You don't need to place submit, cancel, reset buttons on the same column.

- **Why to fix:**

When fields are placed side by side on a magnified screen, one field can fill the screen and thus users are unaware of other fields next to it. For users with low vision, it is easier to fill out a form when all the fields are stacked vertically. One user with low vision said, "It can be inconvenient on a form when you have two entry fields side by side. It's easier to see top to bottom."

- **Learn More:**

- HTML 4.0 standard on [forms in general](#)
- HTML 4.0 standard on the [FORM tag](#)
- HTML 4.0 standard on the [LABEL tag](#)
- a [detailed discussion](#) on linearizing layout tables and its effects on navigation

29 button close to field (43)

29.1 Put 'Go' button close to selection box or entry field [NNg-ACC 43]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- Forms
- forms and fields
- Manual

- **Test Description:**

This test implements guideline # 43 on page 88: "On any page with a single selection box or entry field, put the Go button as close as possible to that box or field".

The test checks if the page contains a form with only one single selection box or an entry field.

If this control is not immediately followed by a submit button, the test asks the user to verify if there is a submit button very close to that box or field.

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

The page contains a form with a single selection box or entry field. Remember to put the Go button as close as possible to that box or field.

- **How to fix:**

Check the distance between the form control (selection box or input field) and the form button (often a "Go" button). If it is possible that a user of a screen magnifier viewing the form is not able to see that there is a button associated to the form, consider moving the button closer to the form control.

You can do that by reducing the white space separating the button from the control. Or by moving the button just below the control.

Try to increase font size of the browser and see if the distance increases and how much. Try to use a screen magnifier to determine under which circumstances the button is not seen together with the control it refers to.

Avoid using a visual layout that separates the form controls and the button. Use the same background color and use visual clues to group the form controls with the button.

If you can, avoid using drop-down menus whose options can be selected without clicking on a "Submit" or "Go" button. These are usually implemented as roll-overs in javascript and not all browsers and screen readers support them.

- **Why to fix:**

With magnified screens, the "Go" button that often accompanies entry fields and selection boxes is sometimes off-screen. Users with low vision would thus pause after entering text or clicking a selection in a drop-down list box, waiting for something to happen. They didn't realize they needed to scroll, find the button, and click it to actually submit their information.

This behavior happens because many users are used to drop-down menus whose options can be selected without clicking on a button.

In one of our studies, a high-tech company, positioned the "Continue" and "Calculate again" buttons too far from the "Choose the quantity" drop-down choice. When the screen is magnified, it is difficult to know the button exists, let alone to find it and invoke an action.

- **Learn More:**

- HTML 4.0 standard on [INPUT tag](#)
- HTML 4.0 standard on [possible types of INPUTs](#)
- HTML 4.0 standard on the [FORM tag](#)
- HTML 4.0 standard on [forms in general](#)
- a [detailed discussion](#) on linearizing layout tables and its effects on navigation

30 submit button (44)

30.1 Put the submit button close to fields [NNg-ACC 44]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- Forms
- forms and fields
- Manual

- **Test Description:**

This test implements guideline # 44 on page 89: "In forms, put the submit button as close as possible to the last field entry box or selection box on the form."

The test checks if the page contains a form with the submit button that is not very close to last entry field or selection box.

In particular the submit button must be at least inside a block element (p, div, td...) that is adjacent to the block element containing the last entry field or selection box.

Otherwise the test asks the user to check that the submit button is as close as possible to the other form elements.

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

The page contains a form with a submit button that is not very close to the last entry field or selection box of the form. Remember to put the submit button as close as possible to the last field or control.

- **How to fix:**

Check the distance between the last form control (selection box or input field) and the submit button. If it is possible that a user of a screen magnifier viewing the form is not able to see that there is a button associated to the form, consider moving the button closer to the form.

Try to increase font size of the browser and see if the distance increases and how much. Try to use a screen magnifier to determine under which circumstances the button is not seen together with the form.

Avoid using a visual layout that separates the form controls and the button. Use the same background color and use visual clues to group the form controls with the button.

- **Why to fix:**

In several cases during our studies, we saw participants with low vision unable to find a form's submit button. After filling out forms and waiting for something to happen, users typically realized that they needed to submit it and would look for a submit button. Sometimes it took them minutes to find it.

On one site, a line spanning a billing form above the "Continue" button made participants using screen magnifiers think that the line indicated the end of the page. They could not find the "Continue" button below it. One user commented, "Continue was below a separator line." This line gave the illusion of completion.

It is also important to make sure that pressing the Enter key will submit the form when a user tabs to the Submit button and presses Enter.

- **Learn More:**

- HTML 4.0 standard on [forms in general](#)
- HTML 4.0 standard on the [FORM tag](#)
- a [detailed discussion](#) on linearizing layout tables and its effects on navigation

31 form instructions (45)

31.1 Put instructions before the field [NNg-ACC 45]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- Forms
- forms and fields
- Manual

- **Test Description:**

This test implements guideline # 45 on page 89: "Put any instructions pertaining to a particular field before the field, not after it".

The test checks if the page contains a form with a selection box or an entry field followed by text.

The test asks the user to verify that this text doesn't contain information on how to fill the control. If this is the case, the test suggests to move these instructions before the field.

This test does not check if the form has LABEL elements properly defined. There is another test, in the basic accessibility packages of LIFT (i.e. Section 508 or W3C/WCAG P.1) that do that.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

The page contains a form with fields or other controls that are followed by text. Remember to put any instructions pertaining to a particular field **before the field**, not after it.

- **How to fix:**

Check if the text following the control or field is an instruction on how to fill-in the field (or use the control).

If this is the case, then try to move the instruction text so that it is read before the form field or control. Put it to the left of the field/control or above it (depending on the form layout).

Alternatively, for cases where putting the text label before the control would lead to confusion (the only case we know of is the conventional layout of checkboxes and radiobuttons, whose labels are placed after the control), use the LABEL element that HTML provides.

In the basic accessibility packages of LIFT (i.e. Section 508 or W3C/WCAG p. 1) there are tests that check if labels are properly defined.

LIFT also offers you powerful Fix Wizards to define labels.

- **Why to fix:**

People listening to screen readers need to hear instructions about fields before they get to the entry box. Once they get to the entry, they will want to proceed to type and won't necessarily know to look past the entry box and for further instructions.

During our study we found, for example, a Japanese governmental site where there is a form with fields that gives instructions after the field. Namely, the note after the field reads: "Write in only single byte alphabet or numbers". It is better to present such information before the input box, so screen reader users will hear the instructions before they begin typing. (Also, this particular field label would be easier to understand if it were not using programming jargon.)

- **Learn More:**

- [HTML 4.0 standard on forms in general](#);
- [HTML 4.0 standard on the FORM tag](#);
- [HTML 4.0 standard on the LABEL tag](#);
- [W3C on labeling form controls](#);
- [a tutorial on accessible forms](#)

32 consider timeout (46)

32.1 Consider how long a timeout is [NNg-ACC 46]

- **Test Type:** Manual

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- Forms
- forms and fields
- Manual

- **Test Description:**

This test implements guideline # 46 on page 91: "Carefully consider how long it will be before a timeout will occur".

This test flags forms that contain complex controls (i.e. controls that might require a significant effort to be operated, like textfields, drop-down menus, selection lists).

The test warns the user to make sure that the form has a timeout that is large enough so that disabled users are able to complete it.

In particular the test is manual because often the timeout is caused by the webserver to which the form is sent. LIFT is unable to test the timeout threshold as this would require dozens of minutes just to wait for a timeout to occur.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

The page contains a form that might be timed out (i.e. such that it cannot be submitted after a certain amount of time is passed).

- **How to fix:**

Check if the webserver implements a timeout for the URL to which the content of the form is submitted. Usually, if the webserver uses a session cookie or session identifier in the URL, this is the case.

Ask your webmaster or network administrator if in doubt.

Sessions (and therefore forms) have a typical timeout of 30 minutes. We recommend sticking to this. If the specific application offered in your site requires a shorter timeout, avoid making it less than 10 minutes.

- **Why to fix:**

During our studies, we saw several instances where people using screen magnifiers were faced with timeout errors while filling out forms. For example, on one ticket shopping site,

users must complete the purchase within five minutes. Timeouts occurred several times when users were trying to order tickets. Sites must give users sufficient time to fill out forms before timeouts occur. We recommend ten to fifteen minutes before a timeout occurs.

- **Learn More:**

- HTML 4.0 standard on [forms in general](#)

33 small text (48)

33.1 Avoid small text [NNg-ACC 48]

- **Test Type:** Automatic
- **Test Categories:**
 - ALL TESTS
 - NNg - beyond ALT text
 - presenting text

- **Test Description:**

This test implements guideline # 48 on page 93: "Do not use very small text for body text".

This test flags pages that contain text in a font size that is smaller than preset thresholds. To determine font size the test looks if the text specifies a CSS style rule with the FONT-SIZE property set to a value that is smaller than a predefined threshold.

The test examines:

- FONT or BASEFONT elements with SIZE attribute smaller than a preset threshold
- inline CSS specifications of elements P, SPAN, DIV, BODY, LI, TD
- and CSS rules contained in STYLE elements in the page
- and CSS rules contained in external CSS files.

The predefined thresholds for FONT/BASEFONT's SIZE attribute are stored in the parameter called "Min. size in FONT/BASEFONT", whose default value is "3".

The predefined thresholds for CSS property "font-size" are stored in the parameter called "Too small font-size (symbols)", whose default value is: small|x-small|xx-small, and the parameter "Min. font-size (pixels)", whose default is 11.

In the former case, if the text has a font-size corresponding to one of the words specified in the parameter "Too small font-size (symbols)", then the test flags the page with an issue.

Similarly, if the text has a font-size specified in points (pt) or pixels (px) whose value is smaller than the one stored in the parameter "Min. font-size (pixels)", then the test flags the page with an issue.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

Page contains CSS rules or FONT elements that specify a font size that is too small. In particular their font size matches the ones specified in the parameter called "Too small font-size (symbols)" (by default set to small|x-small|xx-small) or it the font size is smaller than the value stored in "Min. font-size (pixels)" (by default set to 11).

- **Specific Issues:**

- **FONT/BASEFONT elements with SIZE that is too small:** The FONT (or BASEFONT) element specifies a value of SIZE attribute that is too small. (Consider also that these tags are deprecated by the W3C and that they are not part of HTML anymore.)
- **inline style FONT-SIZE is too small:** The inline CSS style specification contains a FONT-SIZE that is too small.
- **inline style FONT-SIZE is too small and absolute:** The inline CSS style specification contains a FONT-SIZE that is too small and that uses an absolute font size.
- **STYLE element with too small FONT-SIZE:** The page contains a STYLE element with a FONT-SIZE property for text set to a value that is too small. (Note: there is an issue like this one for each FONT-SIZE that is too small.)
- **STYLE element with too small and absolute FONT-SIZE:** The page contains a STYLE element with a FONT-SIZE property for text set to a value that is too small and that is specified with absolute font size. (Note: there is an issue like this one for each FONT-SIZE that is too small.)
- **external CSS file with too small FONT-SIZE:** The page uses an external CSS file with a FONT-SIZE property for text set to a value that is too small. (Note: there is an issue like this one for each FONT-SIZE that is too small.)
- **external CSS file with too small and absolute FONT-SIZE:** The page uses an external CSS file with a FONT-SIZE property for text set to a value that is too small and that is specified with absolute font size. (Note: there is an issue like this one for each FONT-SIZE that is too small.)

- **Test Parameters:**

- **Min. font-size (pixels)** with default value: 11 .
This parameter has to be a number. It represents the minimum size (in pixels) of the CSS property "font-size" for text and links inside the document.
- **Too small font-size (symbols)** with default value: small|x-small|xx-small .
This parameter can be a list of bar-separated words. Each word is a possible value of the CSS property "font-size". (Words are case insensitive).
More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.
- **Min. size in FONT/BASEFONT** with default value: 3 .
This parameter has to be a number. It represents the minimum size of text (specified with the deprecated tags FONT or BASEFONT and their attribute SIZE) of text and links inside the document.
Admissible values for the parameter are numbers 1 to 7.

- **How to fix:**

The easiest way to fix this problem is to enlarge the font size of the text. Change the CSS property that applies to the text and enlarge the font size there.

Consider that the W3C discourages webdesigners to use the FONT tag in favor of a CSS style rule.

If you don't want to create an external style sheet, what you can do is to add the following code within the HEAD tag of your document in order to display all text using the default text size of the browser used by the visitor:

```
<STYLE>
  BODY,TD,P,DIV,SPAN {font-size: normal}
</STYLE>
```

- **Why to fix:**

Use fonts that are at least 11 point, or better yet, let the user control the exact size while you control the relative size. With 11-point text, users with low vision users will still have to zoom to read it, but when the original is that size, there is a good chance that letters will still be readable when magnified. After using a website with text smaller than 11 points, one user with low vision said, "The print should have been bigger. There was too much graphics. The writing was too small and it was hard to notice the links."

In many cases links were small because the style sheet specified 8-point type, making it difficult to read for people with low vision.

When headings and category text was very small or subtle (like a muted gray), users with low vision did not see the headings. Category titles are important for users with low vision as they provide context. Making it impossible for them to see or notice the categories contributes to the existing context issues magnification brings.

- **Learn More:**

- [Alertbox on why using and how to use CSS](#)
- [CSS1 official specifications](#)
- [HTML 4.0 standard on adding styles to documents](#)
- [HTML 4.0 standard on the STYLE element](#)
- [HTML 4.0 standard on the style attribute](#)
- [what the W3C says about browsers default styles](#)
- [a CSS beginners guide](#)
- [browsers compatibility chart](#)
- [detailed CSS browsers known bugs](#)
- [about Fitt's law](#)

34 large tables (68)

34.1 Avoid large tables [NNg-ACC 68]

- **Test Type:** Automatic

- **Test Categories:**

- ALL TESTS
- NNg - beyond ALT text
- Tables

- **Test Description:**

This test implements guideline # 68 on page 110: "Avoid using large tables for any reason. If you must use them, consider providing the information in text as well".

The test checks if the page contains tables used to present data. If so, it checks if tables have more than N rows or M columns.

The default value for N is 15 and for M is 15. N and M are stored in the parameters called "Max number of rows" and "Max number of cols".

This test has **priority 2**: users were very frustrated when this guideline was violated.

- **Issue Description:**

The page contains tables that are too large (according to parameters "Max number of rows" and "Max number of cols").

- **Specific Issues:**

- **too many rows:** The table has more than N rows (by default N = 15).
- **too many columns:** The table has more than M columns (by default M = 15).

- **Test Parameters:**

- **Max number of rows** with default value: 15 .
This parameter can be any number greater than 0. It represents the **maximum number of rows** in a data table.
- **Max number of cols** with default value: 15 .
This parameter can be any number greater than 0. It represents the maximum number of columns in a data table.

- **How to fix:**

Reduce the size of the table. If that is not possible, then try to summarize the table content in a page, and put the table in a secondary more detailed page.

Don't forget to summarize the table and to make it accessible (according to W3C WCAG or Section 508 requirements) by defining header cells, their ID and using the HEADERS attribute to associate data cells to header cells.

If you can, spend some time in going through the LIFT tutorial that explains how to make accessible tables (go to the online help and click on the tutorial section).

- **Why to fix:**

With so much information in a table, it is difficult for screen reader users and Braille users to make sense of it all and remember it. For screen magnifier users it is very difficult to get context and navigate a large table. It is especially difficult to recall which column each cell belongs under, as the column and row headers are not visible when the table is magnified.

- **Learn More:**

- HTML 4.0 standard on [tables in general](#);
- HTML 4.0 standard on the [TH and TD elements](#);
- HTML 4.0 standard on [table rendering](#) by various types of browsers
- W3C/WAI on [how to make accessible data tables](#);
- a [detailed discussion](#) on linearizing tables

35 summarize tables (71)

35.1 Summarize tables [NNg-ACC 71]

- **Test Type:** Automatic
- **Test Categories:**
 - ALL TESTS
 - NNg - beyond ALT text
 - Tables

- **Test Description:**

This test implements guideline # 71 on page 114: "Summarize your tables".

The test checks if the page contains tables used to present data. If so, it checks if the table has the SUMMARY attribute and if it contains some text (at least N words have to be present, where N is the value of the parameter "Min. summary words", by default equal to 3).

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

The page contains tables that don't have the SUMMARY attribute or it appears to be inadequate as a summary of the table content.

- **Specific Issues:**

- **not defined:** The table has no SUMMARY.
- **too few words:** The table has a SUMMARY with too few words to be really informative.

- **Test Parameters:**

- **Min. summary words** with default value: 3 .
This parameter can be any number greater than 0. It represents the **minimum number of words that can be contained in a table summary** to qualify the summary as being not significant.

- **How to fix:**

Use the SUMMARY attribute for specifying a summary of the table content. Remember not to use HTML within the attribute.

You can use the Fix Wizard to view and edit the summary text.

- **Why to fix:**

When listening to a screen reader read a table, users often hear something like: Table has four columns and two rows. It then immediately launches into reading across the first row, depending upon how the developer has coded the table. Since the user can't scan the table all at once, it's difficult to get a sense of what the table contains. Users basically have to interpret what the items in the first row are, and from that get an understanding of what might be in the rows to follow. Even a simple table that is relatively easy to follow can be easier to understand if the developer first summarizes the table.

Many users said they can envision tables, even if they have never actually seen one. When tables are used for layout, or aesthetic purposes, this can especially confuse users, as they expect the table to be used to present data.

- **Learn More:**

- HTML 4.0 standard on [tables in general](#);
- HTML 4.0 standard on [table rendering](#) by various types of browsers
- W3C/WAI on [how to make accessible data tables](#)
- a [detailed discussion](#) on linearizing tables.

36 describe frames (73)

36.1 Describe all frames [NNg-ACC 73]

- **Test Type:** Automatic
- **Test Categories:**
 - ALL TESTS
 - NNg - beyond ALT text
 - Frames
 - tables and frames

- **Test Description:**

This test implements guideline # 73 on page 119: "Describe all frames".

The test checks if the page contains FRAMESET with FRAMEs that do not have the TITLE attribute and/or the LONGDESC attribute.

This test has **priority 1**: users could not complete the task at all, or were significantly frustrated when this guideline was violated.

- **Issue Description:**

The page is laid out using frames, some of which appears to be poorly described, either because their TITLE attribute is not properly defined (it is missing, or it is blank or it contains HTML) or because their LONGDESC attribute is not properly defined (it is missing, it contains an invalid URL, ...).

- **Specific Issues:**

- **TITLE not defined:** Invalid FRAME: missing TITLE attribute of FRAME element
- **TITLE is empty:** Invalid FRAME: TITLE attribute of FRAME element exists but is the empty string ""
- **TITLE is blank:** Invalid FRAME: TITLE attribute of FRAME element exists but is the blank string " "
- **TITLE with html tags:** Invalid FRAME: TITLE attribute of FRAME element exists but contains HTML tags
- **LONGDESC attribute not defined:** Missing LONGDESC attribute of FRAME element.
- **LONGDESC file does not exist:** Invalid LONGDESC attribute of FRAME element: mentioned file does not exist.
- **LONGDESC file is not HTML:** Invalid LONGDESC attribute of FRAME element: mentioned file is not an HTML file.
- **LONGDESC file is empty:** Invalid LONGDESC attribute of FRAME element: no file specified.

- **LONGDESC URL is dead:** Invalid LONGDESC attribute of FRAME element: its URL is dead.
 - **LONGDESC URL generated bad response:** Invalid LONGDESC attribute of FRAME element: got an error response from the server of its URL.
 - **LONGDESC URL has a bad protocol:** Invalid LONGDESC attribute of FRAME element: it is not a local file nor an HTTP URL.
 - **framed page with more than one title:**
TITLE tag is duplicated (different browsers will pick up different titles) in framed page
 - **missing title in framed page:** There is no TITLE tag in framed page
 - **empty title in framed page:**
TITLE of framed page has no content
 - **framed page title contains placeholders:**
TITLE of framed page contains only placeholder text
- **How to fix:**

If you use frames, tell screen reader users how many frames are on a given page. Always summarize every frame so users are aware of the general contents before they decide to listen to the entire frame.

The TITLE attribute should be defined because its value is all that (some) non-graphical browsers show. In fact, each frame will be shown independently for the other ones, making it hard for the user to figure out their relationship. Titles like "content area" or "navigationals" are much more informative than "left" or "top-frame".

For example, the following example shows how to use the TITLE attribute to provide a clearer description of the frame content. It also shows how you can define a page ("frameset-desc.html") that contains different sections, each one describing each frame.

```
<FRAMESET rows="20%,*">
  <FRAME src="promo.html" name="promo" title="promotions"
  longdesc="frameset-desc.html#promo">
  <FRAME src="sitenavbar.html" name="navbar"
  title="Sitewide navigation bar" longdesc="frameset-desc.html#navbar">
</FRAMESET>
```

However be aware that at the moment few browsers support the TITLE or the LONGDESC attributes. Your only chance of helping visitors is to name each frame (using the NAME attribute) with informative names that refer to frames content (like "navigation_bar") rather than to frames positions (like "top_left"). Avoid using the word "frame" in these names as assistive technologies already use the word "frame" when describing the frameset.

Another thing that you should do with frames is to make sure that framed pages (i.e. the pages corresponding to the SRC attribute of the FRAME element) have a well defined TITLE element.

- **Why to fix:**

Frames are one of the commonly known causes of accessibility issues. The biggest problems with frames occurred in our studies with people using screen reader and Braille devices. Basically, such devices sometimes fail to read the content frames. At other times, users see one frame's contents, but that frame eclipses other content that users are unaware of and thus never look for. These are probably understatements of the problem, but as many developers and users know, the issues are severe.

One participant brought to our attention a website of a company that makes natural pet food, which is important for guide dogs with food allergies. This is her account: "I found a website today that is totally inaccessible to the users who are blind who could benefit from it. When I visited the site, the initial screen contained frames, but also some general text describing the company and its products. Each frame (most of them anyway) was appropriately labeled with alternative image text. However, when I entered the frames, I got more frames sometimes the same ones, sometimes a different set. I never could get to the text I wanted to read. The text I wanted was a frame labeled Encyclopedia of Ingredients. There is also another frame containing a side-by-side comparison of ingredients between different brands of dog food. This is the information that I wanted."

In our studies, we had some participants who had low vision but did not use screen magnification software. Those people typically increased font size in their browsers, but the font size only changed in the selected frame. As a result, they had to select the target frame before changing the font size. This was tedious.

- **Learn More:**

- HTML 4.0 standard on [frames](#) ;
- HTML 4.0 standard on the [title attribute](#);
- W3C/WAI [guideline 12](#);
- J. Nielsen's [Alertbox on frames](#);
- a discussion on [creating quality frames](#).

A Test Categories

ALL TESTS This category contains all the available tests of all installed packages.

NNg - beyond ALT text This category contains the tests that implement the guidelines described in the **Nielsen Norman Group** report **Beyond ALT Text: Making the Web Easy to Use for Users with Disabilities**.

links and buttons This category contains tests implementing guidelines dealing with links and buttons that are described from page 65 of the Nielsen Norman Group report **Beyond ALT Text: Making the Web Easy to Use for Users with Disabilities**.

pop-up windows, new windows This category contains tests implementing guidelines dealing with pop-up windows, rollover text, new windows, and cascading menus that are described from page 57 of the Nielsen Norman Group report "Beyond ALT Text: Making the Web Easy to Use for Users with Disabilities".

Images This category contains tests dealing with images.

graphics This category contains tests implementing guidelines dealing with graphics and multimedia that are described from page 42 of the Nielsen Norman Group report: "Beyond ALT Text: Making the Web Easy to Use for Users with Disabilities".

links and buttons This category contains tests implementing guidelines dealing with links and buttons that are described from page 65 of the Nielsen Norman Group report: "Beyond ALT Text: Making the Web Easy to Use for Users with Disabilities".

page organization This category contains tests implementing guidelines dealing with page organization that are described from page 73 of the Nielsen Norman Group report: "Beyond ALT Text: Making the Web Easy to Use for Users with Disabilities".

forms and fields This category contains tests implementing guidelines dealing with forms and fields that are described from page 82 of the Nielsen Norman Group report: "Beyond ALT Text: Making the Web Easy to Use for Users with Disabilities".

presenting text This category contains tests implementing guidelines dealing with presenting text that are described from page 92 of the Nielsen Norman Group report: "Beyond ALT Text: Making the Web Easy to Use for Users with Disabilities".

tables and frames This category contains tests implementing guidelines dealing with tables and frames that are described from page 110 of the Nielsen Norman Group report: "Beyond ALT Text: Making the Web Easy to Use for Users with Disabilities".

Links This category contains tests implementing guidelines dealing with link objects.

Scripts This category contains tests implementing guidelines dealing with script objects.

cascading menus This category contains tests implementing guidelines dealing with cascading menus.

Imagemaps This category contains tests implementing guidelines dealing with image maps (aka hotspots).

Title This category contains tests implementing guidelines dealing with the TITLE element.

Tables This category contains tests implementing guidelines dealing with tables.

Forms This category contains tests implementing guidelines dealing with forms.

Frames This category contains tests implementing guidelines dealing with frames.

Manual This category contains tests implementing guidelines that require further human investigation to determine if the reported issues are true.

generic guidelines This category contains tests implementing guidelines that require further human investigation to determine if the reported issues are true and that apply to whole site.

multimedia This category contains tests implementing guidelines dealing with multimedia objects.

A Test Parameters

Max number of images This parameter can be any number greater than 0. It represents the **maximum number of images** (except spacers) that can be contained in a page.

Default value: 15

Affected Tests:

- Minimize use of graphics [NNg-ACC 02]

Max number of links This parameter can be any number greater than 0. It represents the **maximum number of links** that can be contained in a page.

Default value: 20

Affected Tests:

- Avoid too many outgoing links [NNg-ACC 17]

Min. distance (chars) This parameter can be any number greater than 0. It represents the **minimum number of text characters** that have to be placed between adjacent text links in order to consider the links sufficiently well separated.

Default value: 3

Affected Tests:

- Leave space between textual links [NNg-ACC 19]
- Leave space between buttons [NNg-ACC 19]

Min. horiz. distance (pixels) This parameter can be any number greater than 0. It represents, in pixels, the **minimum width** of images that have to be placed between horizontally adjacent links in order to consider the links sufficiently well separated.

Default value: 25

Affected Tests:

- Leave space between textual links [NNG-ACC 19]
- Leave space between buttons [NNG-ACC 19]

Min. vert. distance (pixels) This parameter can be any number greater than 0. It represents, in pixels, the **minimum height** of images that have to be placed between vertically adjacent links in order to consider the links sufficiently well separated.

Default value: 10

Affected Tests:

- Leave space between buttons [NNG-ACC 19]

Min. link length (chars) This parameter can be any number greater than 0. It represents the **minimum length** (in characters except spaces) of the label of a textual A link (i.e. its content) or of a textual button.

Default value: 6

Affected Tests:

- Avoid too short text links or button labels [NNG-ACC 18]

Min. link width (pixels) This parameter can be any number greater than 0. It represents the **minimum width** of the image that is the label of an A link (i.e. its content).

Default value: 25

Affected Tests:

- Avoid too small buttons [NNG-ACC 18]
- Avoid too small hotspots [NNG-ACC 18]

Min. link height (pixels) This parameter can be any number greater than 0. It represents the **minimum width** of the image that is the label of an A link (i.e. its content).

Default value: 25

Affected Tests:

- Avoid too small buttons [NNG-ACC 18]
- Avoid too small hotspots [NNG-ACC 18]

Max number of rows This parameter can be any number greater than 0. It represents the **maximum number of rows** in a data table.

Default value: 15

Affected Tests:

- Avoid large tables [NNg-ACC 68]

Max number of cols This parameter can be any number greater than 0. It represents the maximum number of columns in a data table.

Default value: 15

Affected Tests:

- Avoid large tables [NNg-ACC 68]

Max number of entries This parameter can be any number greater than 0. It represents the **maximum number of menu entries** in a SELECT element.

Default value: 10

Affected Tests:

- Avoid long cascading menus [NNg-ACC 16]

Window return label This parameter can be a bar separated list of words. It represents the **possible words that can be labels of a link or button** that closes a window or that returns to previous page. (Words are case-insensitive.)

More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

Default value: close|return|previous

Affected Tests:

- Provide easy ways to go back when a new window is opened [NNg-ACC 14]

Return to HOME label This parameter can be a bar separated list of words. It represents the **possible words that can be labels of a link or button**. These links or buttons are then assumed to lead to the site's home page. (Words are case-insensitive.)

More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

Default value: home

Affected Tests:

- Provide easy ways to get back to the site's home page [NNg-ACC 14]

Logo file This parameter has to be a filename. It represents part of the pathname of the file containing the company logo. It is used to identify the logo in a page: any image whose SRC attribute contains this filename is supposed to be the logo.

Default value: logo.gif

Affected Tests:

- Do not associate the word 'homepage' with your company logo [NNg-ACC 26]

Words in logo ALT This parameter can be a bar separated list of words (regular expressions). It represents the mandatory part of the ALT associated to the logo image, if any. The logo file is identified by its name, stored in the parameter "Logo file".

More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

Default value: `link to|go to|open|view`

Affected Tests:

- Do not associate the word 'homepage' with your company logo [NNg-ACC 26]

Invalid title content This parameter can be a bar separated list of words. It represents the **possible words that can be contained in the page title** to qualify the title as being not significant. (Words are case-insensitive.)

More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

Default value: `untitled`

Affected Tests:

- Confirm what the page is: check its title [NNg-ACC 25]

Min. summary words This parameter can be any number greater than 0. It represents the **minimum number of words that can be contained in a table summary** to qualify the summary as being not significant.

Default value: `3`

Affected Tests:

- Summarize tables [NNg-ACC 71]

ALT placeholders This parameter can be a bar separated list of words. When an ALT of an image is analyzed, LIFT checks if the ALT contains any of these words. If so, LIFT assumes that the ALT contains only a placeholder, not significant text.

More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

Default value: `\.gif|\.jpg|\.jpeg|\.jpe|\.png|bytes`

Affected Tests:

- Use ALT text to describe images [NNg-ACC 03]

Max ALT length (chars) This parameter has to be a number. It represents the maximum number of words that may appear in the ALT attribute of an image.

More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

Default value: `150`

Affected Tests:

- Use ALT text to describe images [NNg-ACC 03]

Min. size in FONT/BASEFONT This parameter has to be a number. It represents the minimum size of text (specified with the deprecated tags FONT or BASEFONT and their attribute SIZE) of text and links inside the document.

Admissible values for the parameter are numbers 1 to 7.

Default value: 3

Affected Tests:

- Avoid small text [NNg-ACC 48]

Min. font-size (pixels) This parameter has to be a number. It represents the minimum size (in pixels) of the CSS property "font-size" for text and links inside the document.

Default value: 11

Affected Tests:

- Avoid tiny links [NNg-ACC 18]
- Avoid small text [NNg-ACC 48]

Too small font-size (symbols) This parameter can be a list of bar-separated words. Each word is a possible value of the CSS property "font-size". (Words are case insensitive).

More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

Default value: `small|x-small|xx-small`

Affected Tests:

- Avoid tiny links [NNg-ACC 18]
- Avoid small text [NNg-ACC 48]

Max num. of text fields This parameter represents the maximum number of active text fields that can be present in a form.

Default value: 10

Affected Tests:

- Limit amount of information required by forms [NNg-ACC 35]

Max num. of form controls This parameter represents the maximum number of active text fields that can be present in a form.

Default value: 20

Affected Tests:

- Limit amount of information required by forms [NNg-ACC 35]

Min. width/height of significant images This parameter represents the minimum width or height that an image has to have in order for LIFT to consider it an image that can convey information.

Default value: 50

Affected Tests:

- Use LONGDESC to describe images in detail [NNg-ACC 03]
- Provide information also in text [NNg-ACC 05]
- Refer to alternate ways for getting graphics information [NNg-ACC 06]
- Avoid using shrunk-down pictures of other pages [NNg-ACC 07]
- Use crisp and clear images [NNg-ACC 08]

Pattern for 'text only' links This parameter represents the content of textual links or of ALT of images embedded within links. When this content pattern is found within these elements, the link is assumed to lead to a text-only version of the page.

More precisely the value of the parameter can be any javascript regular expression, that is compiled with case insensitive mode.

Default value: `(text\s*only)|(only\s*text)|(text\s*version)`

Affected Tests:

- Avoid automatically created text-only pages [NNg-ACC 10]