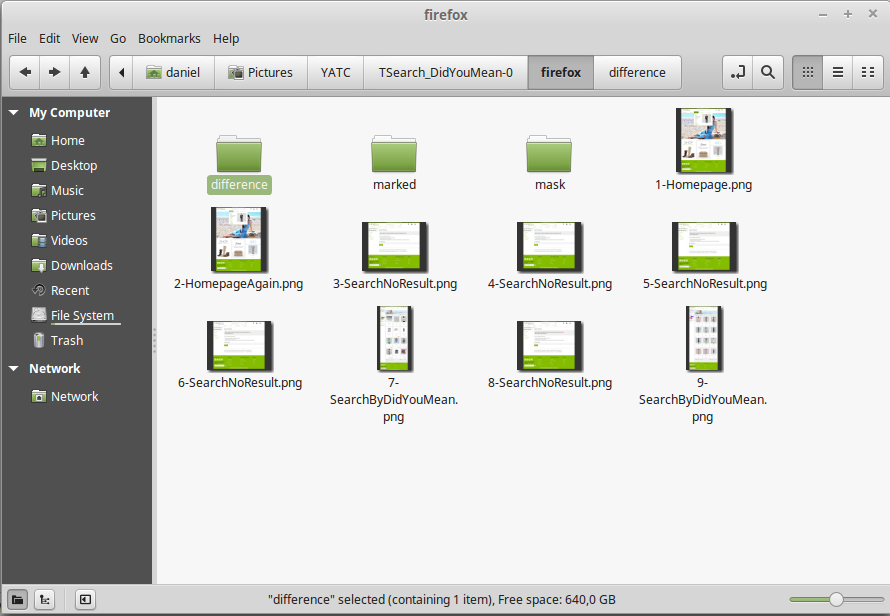
# Screenshot Comparison User Guide:

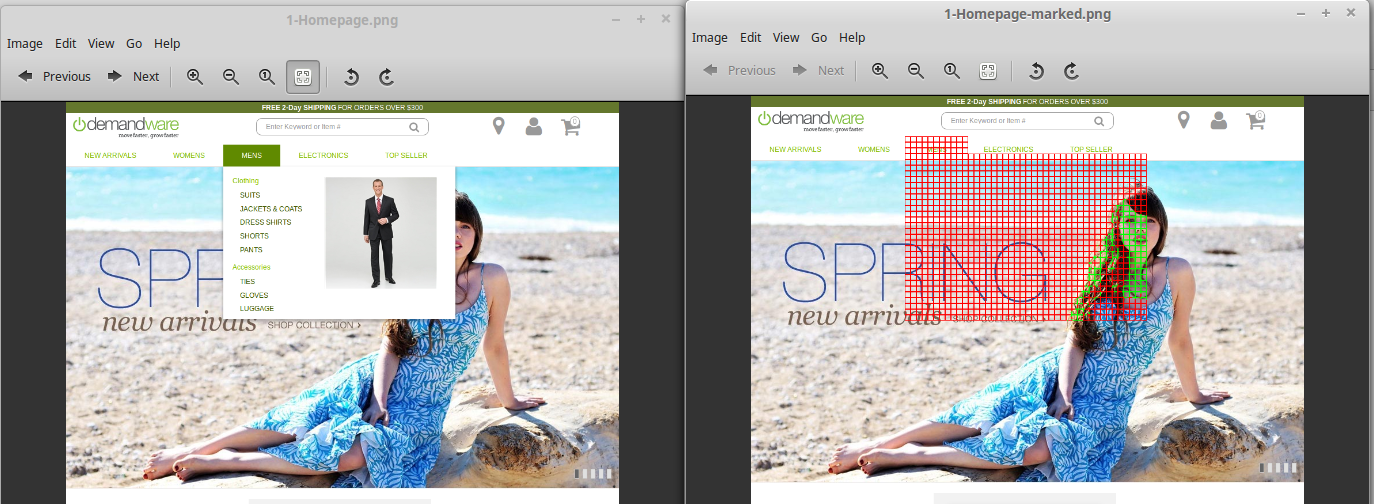
# Why Screenshot Comparison:

The screenshot comparison tests the layout of the site. The layout is dependent on many factors and css values. Instead of checking them all, the screenshot comparison tests what the user actually sees. It checks the result without looking at technical values.

# What does it do:

When you run the screenshot comparison, it creates a folder with the name of the testcase, a subfolder with the name of the browser and the taken images inside the folders.

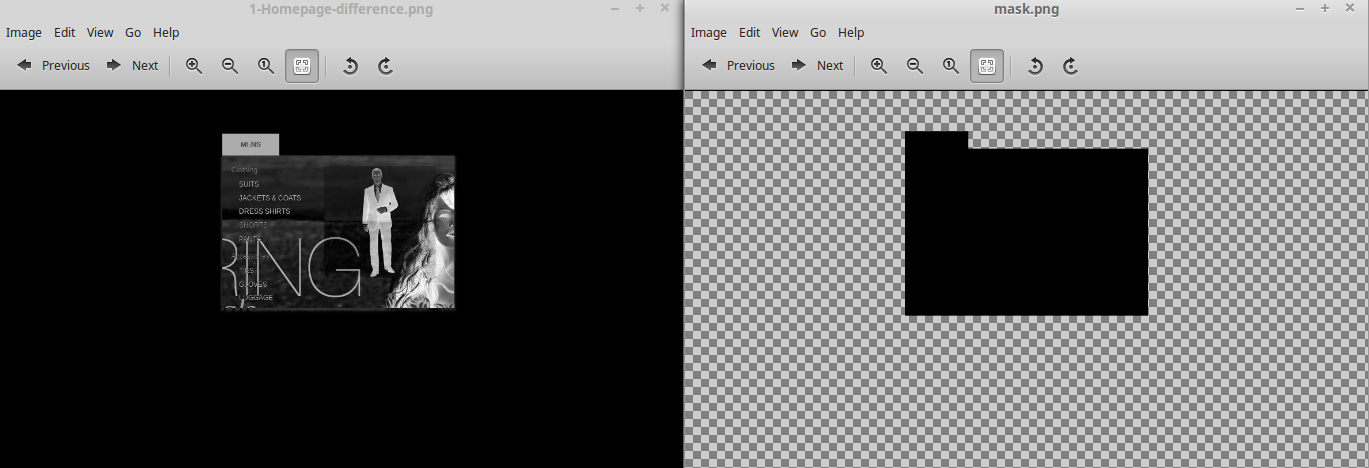


The names of the images consist of an index and the name of the action they were taken in. If pictures with these names already exist they will be used as reference pictures for the image comparison. If there are no differences, nothing happens. If there are differences, the image is saved in the marked folder. Also, the differences are marked and the resulting image is also saved in the marked folder. The test case will fail with an assertion. The same happens if the websites size changed.

The screenshot comparison can be set to create a difference image in addition to the marked image. The difference image is a greyscale image where the brightness depends on the color difference at that point. The bigger the difference, the brighter the pixel.

The screenshot comparison can also be set to run in training mode. In that case, differences will be masked, not marked. This means the program will create a mask image for every screenshot. Detected differences will be painted black in the mask image and black areas are ignored in later comparisons. If more differences are detected in later training runs, they will also be painted black. This means additional training runs can further mask the image, they can never unmask it. Masked areas are painted black in the marked image.

It is possible to manually edit the mask image. You can set areas black in the mask image to mask them or set them transparent to unmask them.



### Properties:

The possible properties and their default values are:

*com.xceptance.xlt.imageComparison.waitTime = 300 com.xceptance.xlt.imageComparison.markBlockX = 10 com.xceptance.xlt.imageComparison.markBlockY = 10 com.xceptance.xlt.imageComparison.directory = no default com.xceptance.xlt.imageComparison.pixelPerBlockXY = 20 com.xceptance.xlt.imageComparison.colTolerance = 0.1 com.xceptance.xlt.imageComparison.pixTolerance = 0.2 com.xceptance.xlt.imageComparison.trainingMode = false com.xceptance.xlt.imageComparison.closeMask = false com.xceptance.xlt.imageComparison.closeWidth = 5 com.xceptance.xlt.imageComparison.closeHeight = 5 com.xceptance.xlt.imageComparison.differenceImage = true com.xceptance.xlt.imageComparison.algorithm = FUZZY*

They will be explained by functionality.

### Fuzzyness parameters:

*com.xceptance.xlt.imageComparison.algorithm = FUZZY*

The pixelPerBlockXY, colTolerance, pixTolerance and algorithm parameters determine how strictly the screenshots should be compared.

The possible values of the algorithm property are EXACTLY, PIXELFUZZY and FUZZY.

With EXACTLY, any displacements and any differences in color will be detected. With PIXELFUZZY, all displacements will still be detected, but small differences in color won't be detected. With FUZZY, small displacements and small differences in color will be tolerated.

*com.xceptance.xlt.imageComparison.colTolerance = 0.1*

The colTolerance property provides a way to account for small changes in color when it comes to large objects or background color. It should be between 0 and 1, a 0.1 stands for a tolerance of 10%, a 1 for 100%. The colTolerance property is not effective when it comes to displacements or high contrasts like those in text or some pictures. Please note that some websites use different shades of the same color to divide their site into different blocks. If the colTolerance value is to high, differences there would not be detected.

If the colTolerance value is very high, blatant changes may go undetected. To illustrates which colors result in what color difference, see the illustrations at the end of the manual

*com.xceptance.xlt.imageComparison.pixTolerance = 0.2*

*com.xceptance.xlt.imageComparison.pixelPerBlockXY = 20*

The pixelPerBlockXY and pixTolerance provide a way to tolerate shifts from small objects, especially text. The pixelPerBlockXY value divides the images into squares with a width and height of pixelPerBlockXY. The pixTolerance value stands for the percentage of different pixels in a block. If there are less differences then specified, they won't be marked: If pixTolerance is 0.2 and 19% of the pixels in a block are different, they won't be marked. If 21% of the pixels in a block are different, they will be marked.

For another example, a pixelPerBlockXY value of 20 means 20\*20 blocks, resulting in 400 pixels per block. With a pixTolerance value of 0.01, that means if there are four or less differences in the block, these will not be marked. If there are more differences, they will be marked.

The pixelPerBlockXY, colTolerance, pixTolerance and algorithm properties determine the strictness of the comparison.

# Example:

*com.xceptance.xlt.imageComparison.colTolerance = 0.1*

*com.xceptance.xlt.imageComparison.pixTolerance = 0.2*

*com.xceptance.xlt.imageComparison.pixelPerBlockXY = 20*

*com.xceptance.xlt.imageComparison.algorithm = PIXELFUZZY*

The algorithm is PIXELFUZZY. The PIXELFUZZY algorithm uses the colTolerance value, but not the pixTolerance value and the pixelPerBlockXY. They are ignored.

The colTolerance value is 0.1, differences which are below 10% or less according to the underlying mathematical functions will be ignored. This may remove some rendering variance.

### TrainingMode properties:

*com.xceptance.xlt.imageComparison.trainingMode = false*

If the trainingMode property is true, the screenshot comparison will do a training run. Differences will be masked, not marked and the test case will not fail due to an assertion.

In future runs, the mask will be put over both the reference screenshot and the new screenshot.

Differences inside the masked areas will not be detected. Masked areas will appear black in the marked image.

*com.xceptance.xlt.imageComparison.closeMask = false*

If the closeMask property is true, small gaps in the maskImage will be filled. For example, you test a homepage with dynamically created pictures of products. You let the screenshot comparison run over it with training mode on. Now the products pictures are roughly marked, but only roughly. There are likely some pixels that have the same color, for example a red pair of shoes and a brown pair of shoes with some red stripes. If the closeMask property is true, gaps in the maskimage will be filled.

*com.xceptance.xlt.imageComparison.closeWidth = 5*

*com.xceptance.xlt.imageComparison.closeHeight = 5*

The closeWidth and closeHeight properties determine how big a gap will be filled. If they are high, bigger gaps will be filled. As a rule of thump, the closeMask and closeHeight properties form a rectangle. If there is a difference on every corner of the rectangle (or more), it will be closed.

The closing of the mask image may take some time if the image is big and the closeWidth and closeHeight properties are high.

### Other properties:

*com.xceptance.xlt.imageComparison.waitTime = 100*

The screenshot comparison will wait that many milliseconds before doing anything. This is used to make sure the website has fully loaded before a screenshot is made.

*com.xceptance.xlt.imageComparison.markBlockX = 10 com.xceptance.xlt.imageComparison.markBlockY = 10*

These parameters determine the width and height of the blocks used for marking and masking. While the size of the blocks during marking is purely cosmetic, for masking it makes for a basic fuzzyness in addition to the closemask parameters.

*com.xceptance.xlt.imageComparison.directory = no default*

The directory in which the screenshots are saved. Inside this, folders with the names of the test cases will be created.

*com.xceptance.xlt.imageComparison.differenceImage = true*

If true, a difference image will be saved in the difference image folder. Differences that were tolerated using the fuzzyness parameters will not be written into the difference image.

### Noteworthy tips, tricks and interactions:

The firefox webdriver takes screenshots of the full website, the chrome driver does not.

The chrome webdriver tends to be faster, but needs a bit more waitTime to take stable screenshots.

Differences in size cannot be masked. Therefore, it does not make sense to use the screenshot comparison with a firefox driver on a website that changes it's size very often.

It will take multiple training runs or manual intervention to mask the correct area, if there are dynamic elements with changing size, for example pictures of products on a constant backgound color..

The taken reference screenshots cannot easily be transferred from one operating system or one browser to the other. Differences in the rendering often result in screenshots with different sizes, minor differences in the content and shifts. The same can be true after browser upgrades.

If a certain comparison fails again, previously made difference images, marked images and new screenshot images will be overwritten.

The mouse should be kept at the same place or outside the website window, mouseover popups will ruin the comparison.

Even small changes in text, including formatting, font, size and style can't be accounted for using the fuzzyness parameters.

The pixelPerBlockXY parameter determines the size of the blocks. Their position is determined by their size (first block in the top left corner, the second block below it …). Therefore, the pixelPerBlockXY implicitly determines the blocks position. When it changes, their position shifts, which may result in changes.

### Illustrations:

This section contains images which may help you understand how to use the fuzzyness properties.

In the following pictures, the background color was compared to the colors in the blocks. The difference was printed on to the blocks.

