How to use Scroller 1.4

The scroller component is very easy to use. It can be customized in almost any way. In this documentation you will learn how to use and customize the scroller component to e.g. a timeline describing a company's history or a panorama. You will also be able to use it as a navigation tool. You can download and include it on your non-commercial homepage for free.

There are no limitations in the not-registred version.

If you register your version the label "Not registred - ©nessal, Denmark" will be removed.

Contens

What is the scroller component ?

The scroller component is an applet which creates a canvas with a size given by the applets parameters. If the canvas is bigger than the applet size it is possible to scroll the canvas by dragging it with the mouse. It is possible to have the canvas reflected or let it be repeated. This way it is possible to display a panorama picture describing a 360 degree scenery. Otherwise it can be reflected at the ends.
The canvas is sensitive to the speed of the mouse when dragging, and if the mouse button is released while dragging, the canvas will continue to scroll for a while, depending of the speed the mouse has when the button was released. It is therefore possible to "throw" the canvas in a sudden direction.

The following elements can be inserted on the canvas:

- Pictures
- Text fields (with selected colors and fonts)
- Sound areas that play an au-file each time the cursor enters the area
- Hyperlink areas which by clicking in them works the same way as ordinary hyperlinks.

You can control the scroller component from a javascript program, and you can invoke javascript from the component.

How to use Scroller

Download and extract the "scroller14.zip" (or "scroller14.jar") file.

In each of the example folders there are an example. The examples is activating by open the "exampleX.html"-file in a browser, where X is the number of the example.

Example 1 - Scroller basic

The first example is a timeline describing the Maersk company history. It is started by loading example1.html (placed in the example1 folder) in a browser.

When the example start the canvas runs by itself. This mode is called autorun. First time you click on the canvas with your mouse the canvas will stop. Your can control the position of the canvas by dragging the canvas (hold the mouse button down while you move the mouse). If you release the mouse button while your are dragging, the canvas will continue to run for a while depending of the speed it has, when the button was released.

The example1.html contains the following lines:

```
<HTML>
<HEAD>
   <TITLE>Maersk history</TITLE>
</HEAD>
<BODY BGCOLOR="#FFFFFF">
&nbsp;
<TABLE>
        <TR>
                <TD WIDTH="20">
                </TD>
                <TD><H1><FONT FACE="Arial,Helvetica">The history of Maersk</FONT></H1>
                        <FONT FACE="Arial,Helvetica"><FONT SIZE=-1>You can stop and "throw the
                        canvas around" with the mouse.</FONT></FONT>
                </TD>
        </TR>
</TABLE>
<APPLET codebase="../" code="scroller.class" archive="scroller.jar" width="100%" height="300">
<PARAM name="inputFile" value="example1.txt">
<PARAM name="delimeter" value="#">
<PARAM name="imgwidth" value="7000">
<PARAM name="imgheight" value="300">
<PARAM name="sleeptime" value="20">
<PARAM name="bgColor" value="ffffff">
<PARAM name="reflectx" value="true">
<PARAM name="reflecty" value="true">
<PARAM name="faktorAuto" value="1">
<PARAM name="faktor" value="5">
<PARAM name="test" value="false">
<PARAM name="startincx" value="-1">
<PARAM name="startincy" value="0">
<PARAM name="xpos" value="0">
<PARAM name="ypos" value="0">
<PARAM name="waitTimeDrag" value="300">
Your browser don't support Java. <BR> You can download a new browser from <A
REF="http://www.netscape.com">Netscape</A>
or <A HREF="http://www.microsoft.com">Microsoft</A>&nbsp;
</APPLET>
</BODY>
</HTML>
```

The parameters to the applet has the following meaning:

Parameter	Description	Default value
inputFile	The name of the file describing the context of the canvas	input.txt
delimeter	A character delimiting the varies part of a line in the inputFile. Hint: Don't use '¤' found on the 4-key on windows platforms. Netscape Navigater for Mac cannot understand this charecter.	"#"
imgwidth	The width of the canvas in pixels	7000
imgheight	The height of the canvas in pixels	300
sleeptime	A number describing how long the applet should sleep between each iteration. This number has nothing to do with the speed of the applet (see later), and normally you can omit it.	50
bgColor	Background color of the canvas in RGB-hexadecimal values.	FFFFFF (White)
reflectx	If this parameter is true, the canvas will reflect when it reach the end in the x direction (horizontal), if false the image will start over again (be repeated). *	true
reflecty	If this parameter is true, the canvas will reflect when it reach the end in the y direction (vertical), if false the image will start over again (be repeated) *	true

faktorAuto	An integer describing the speed of the canvas, when it run in "Auto mode".	1
faktor	An integer describing the speed of the canvas, when it is thrown by the mouse	5
test	When building up the layout of the canvas you normally have a large area where you want to place text, images etc. To help you choose the right location on the canvas, the position on the canvas will be display in the browsers statusline when this parameter is true.	false
startincx	An integer describing the direction of the applet in x-direction when it is running in "Auto mode". -1=right-to-left, 0= no Auto run, 1=left-to-right	1
startincy	An integer describing the direction of the applet in y-direction when it is running in "Auto mode". -1=bottom-to-top, 0= no Auto run, 1=top-to-bottom	0
xpos	The start x-position of the canvas. You are responsible that the position is inside the image	0
ypos	The start y-position of the canvas. You are responsible that the position is inside the image	0
mouseControl	If true you can control the canvas with the mouse - otherwise not	true
waitTimeClick	Indicate the time in milliseconds the canvas will stop after a links has been clicked until it starts again	0
waitTimeDrag	Indicate the time in milliseconds the canvas will stop after the canvas has been draged until it starts again. If the value is -1, the canvas will stop permanently.	-1

*) Please note that reflectx and reflecty cannot be false both at the same time. Repetition of the image in both directions is not yet supported.

Note that in the same folder as contains 'example1.html', there are two class-files, 'scroller.class' and 'element.class'. It is important that these files are placed in the same folder as the html-file which contains the applet-tag.

The context of the canvas is controlled by the "example1.txt"-file. You can open it in a text-editor. Each item on the canvas is associated with a line in this file. When the scroller component starts it read the file and the first character in each line decide whether the line describe an image, a text-field, a hyperlink area or a mouse-over field:

0 = Image
1 = Textfield
2 = Hyperlink area
3 = Mouse over/au-sound
4 = Direction information (Example 4)
5 = Wait zone (Example 4)
6 = Jump zone (Example 4)

7 = New URL zone (Example 4)
8 = drawRect zone (Example 4)

No. 0-3 is described in the following the rest wil be explained in example 4.

Please note that blank lines are ignored and that lines can be comment out with two slash (//) in the beginning of the line (like in C++/Java).

The file is read from top to button, and the items is placed on one another on the canvas in this order. You can then put images on top of each other or make text remarks on an image etc.

0 = Image

The first line in example1.txt is:

```
0#50#20#images/m01.jpg
```

The meaning of each of the values between the delimiters symbol '#' are explained below:

Value	Description
0	Number indicating that the line is a picture
50	x-coordinate of the top left of the image
20	y-coordinate of the top left of the image
images/m01.jpg	Relative path to the image according to the class-file location

1 = Text field

The fourth line in example1.txt is

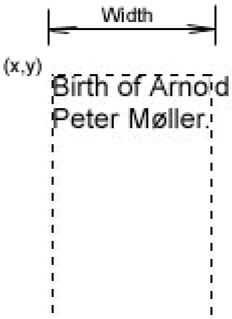
```
1#150#35#"Ariel", 0, 12#000000#150#15#Birth of Arnold Peter Møller.
```

The meaning of each of the values between the delimiters symbol '#' are explained below:

Value	Description
1	Number indicating that the line is describing a text line
100	x-coordinate of the top left of the text line
35	y-coordinate of the top left of the text line
"Ariel", 0, 12	Font for an item: "Font",style,size **Font:** This is the font name of the font. The number of fonts varies greatly

	across systems. If the chosen font is not available, the nearest will automatically be used. Normally at least following fonts is available: TimesRoman, Courier, Helvetica, Dialog, DialogInput **Style:** 0 = normal 1 = italic 2 = bold **Size:** Size of the font
0000ff	Text color in RGB hexadecimal values
250	width of text field in pixels -se explanation below
15	Distance between text lines in pixels. If you write 'null', the component will automatic use an appropriate distance. If you just want the default value, which depends on the font and size, you just write null, and the scroller component will automatically find out which distance to use.
Birth of Arnold Peter Møller.	Text to be displayed

The text field is a bit tricky. Below is showed a text filed which start at (x,y) and have a 'Width'.



The scroller applet will break the text line when a word goes out of the text field, and display the next word on the next line. It is not guarantied that the text will be within the text field. The text field is arrange this way to ensure that all word will be displayed, regardless how long they are. You can force a line break with a /n with space on each side e.g.:

```
1#150#35#"Ariel", 0, 12#000000#150#15#Birth of \n Arnold \n Peter \n Møller.
```

Note that a text filed has no height. It will be filled out as long as there are more words.

2 = Hyperlink area

A hyper link area is an area, in which you can click to activate a hyper link or to invoke a javascript function. There are no hyper links in example1.html, but example2 (in example2.txt) contains the following line

2#20#500#250#520#http://www.microsoft.com#main

- which is a hyper link to Microsoft homepage. The meaning of each of the values between the delimiters symbol '# are explained below:

Value	Description
2	Number indicating that the line is describing an hyper link area
20	x-coordinate of the top left corner of the hyper link area
500	y-coordinate of the top left corner of the hyper link area
250	x-coordinate of the buttom right corner of the hyper link area
520	y-coordinate of the buttom right corner of the hyper link area
http://www.microsoft.com	Url of the hypertext area. If the url doesn't start with 'http' it is assumed to be a relative url to the class files. You can call a javascript routine from the component by specify a hyper link starting with javascript and the function name separated by a colon e.g.. *javascript:displayAlert('Warning')* **Important:** To call javascript from the component you have to place the netscape-folder and sub folder under your class files location. Also if you only intend to use MS Explore browser. Furthermore you have to change the applet tag to include a mayscript-statement e.g.. `<APPLET codebase="." code="scroller.class" width="100%" height="300" MAYSCRIPT` This is an permission from you as an html-author that a component you include in your html document have the permission to call other scripts on your site. - se more in example2.
main	Target frame. If no target frame is chosen, just write null

3 = Mouse over area

A mouse over area is an area where you invoke a javascript function when the mouse enters the area and invoke another javascript function when the mouse leaves the area. There are no mouse over areas in example1.html, but example2 (in example2.txt) contains the following line

3#20#560#250#590#mouse('mouseover.html'#0

To activate example2, load example2.html (in folder example2) into your browser. Example2 is a frame based example. The scroller component itself is contained in scroller.html. Open this file in a text editor to see what it contains.

The meaning of each of the values between the delimiters symbol '#' are explained below:

Value	Description
3	Number indicating that the line is describing an mouse over area
20	x-coordinate of the top left corner of the hyperlink area
560	y-coordinate of the top left corner of the hyperlink area
250	x-coordinate of the buttom right corner of the hyper link area
590	y-coordinate of the buttom right corner of the hyper link area

mouse('mouseover.html')	Name of javascript function to be invoked when the mouse enters the area. The function must be included in the html-page containing the component. In this case the html-file contains a JavaScript function 'mouse':
	`function mouse(s)`
	`{`
	`    parent.main.location.href=s;`
	`}`
	When the mouse leaves the area a javascript function with the same name, but with a preceding 'no_' is called. It is assumed that both functions are included in the html-file containing the component. In this case it is assumed that the following function is included (not necessarily doing the same):
	`function no_mouse(s)`
	`{`
	`    parent.main.location.href="no_"+s;`
	`}`
	You can also choose to specify a java sound file with the .au extension to be played when the user enters the area. Simply write the name of the au-file you want to be played and the component will know by the au extension. Please note that the au. extension not is appropriate for larger music peaches. In these cases midi, wave and other formats is much more convenient - see later how you use these formats.
	Important: To call javascript from the component you have to place the netscape-folder and sub folder under your classifies location. Also if you only intend to use MS Explore browser. Furthermore you have to change the applet tag to include a mayscript-statement e.g..
	`<APPLET codebase="." code="scroller.class"`
	`width="100%" height="300" MAYSCRIPT`
	This is an permission from you as an html-author that a component you include in your html document have the permission to call other scripts on your site.

	- se more in example2.
0	This indicate that the mouse over take effect the first time the mouse enters the area. If you put 1 instead the first time activation will be skipped.

Example 2 and 3 - Interacting with JavaScript

In example 2 and 3 we use javascript to call the scroller component, and we call javascript from the scroller applet to invoke sound hyper links etc. Why and how this is done is explained in this section.

Java and javascript is not the same: A Java program is compiled to a executive .class file just as an program on the windows platform is compiled to a executive .exe-file. The only difference between the two programs is that you have to have a special component to view the .class file called a "Java virtual machine". This component is by standard included in most browsers. Java programs called applet can then be showed directly in the browser.

Javascript is code included directly in the html-document. It is interpreted when the html-document reach the client browser. Most browsers have javascript interpreters.

Beside these differences between Java and javascript there are also differences in what they are used for:

Java is used where you want the ultimate graphic control over an area in the browser. Furthermore you can use Java to make network connection, and even read and write to the users hard disk if you use signed applets. This cannot be done from javascript. You don't use Java to control the browser. To do this you use javascript by which you can control most things in a browser.
As we shall see in the examples to come, javascript and java can communicate with each other - both ways. This way it is possible to do almost anything in a browser, that you normally only expect programs written to a particular platform can do.

Below is shown panorama.html from example3. It is displayed in the main frame to the left when example3.html is loaded into your browser.

```
<html>
<head>
   <meta name="Author" content="nessal/Scroller">
   <title>Panorama</title>
<script LANGUAGE="javascript">


var sound=""

function getSound()
{
   return sound;
}

function play(s)
{
    sound=s;
    parent.sound.location.href="sound.html";
}
```

```
function no_play()
{
    parent.sound.location.href="nosound.html";
}

function jump(x,y)
{
    document.scroller.setPos(x,y);
}

function stop()
{
    document.scroller.stopThread();
}

function start()
{
    document.scroller.setStartRun(true);
    document.scroller.startThread();
}



function getCoords()
{
   xpos=document.scroller.getX();
   ypos=document.scroller.getY();
   x=document.scroller.getCursorX();
   y=document.scroller.getCursorY();
   alert(xpos + " : " + ypos + " : "+ x + " : " + y);
}


function setFactor(x,y)
{
    document.scroller.setFactor(x,y);
}

function setStartInc(x,y)
{
    document.scroller.setStartInc(x,y);
}

</script>
</head>
<body bgcolor="#FFFFCC">
&nbsp;
<table>
<tr>
<td WIDTH="20"></td>
<td>
<h1>
<font face="Arial,Helvetica"><font color="#000066"><font size=+2>Golden
Gate</font></font></font></h1></td></tr>
</table>

<applet codebase="../" code="scroller.class" archive="scroller.jar" name="scroller" width="600"
height="220" MAYSCRIPT>
<param name="inputFile" value="panorama.txt">
<param name="delimeter" value="#">
<param name="imgwidth" value="1912">
<param name="imgheight" value="240">
<param name="sleeptime" value="20">
<param name="bgColor" value="00ff00">
<param name="reflectx" value="false">
<param name="faktorAuto" value="1">
```

```
<param name="faktor" value="5">Your browser don't support Java.
<br>You can download a new browser from <a href="http://www.netscape.com">Netscape</a>
or <a href="http://www.microsoft.com">Microsoft</a>
</applet>

<br><font face="Arial,Helvetica"><font size=-1><a href="JavaScript:play('ironic.mid')">Play
music (ironic - Alanis Morisette)</a></font></font>
 <font face="Arial,Helvetica"><font size=-1><a
href="JavaScript:jump(100,10)">Jump</a></font></font>
<br> <font face="Arial,Helvetica"><font size=-1><a
href="JavaScript:stop()">Stop</a></font></font>
 <font face="Arial,Helvetica"><font size=-1><a href="JavaScript:start()">Start</a></font></font>
<br> <font face="Arial,Helvetica"><font size=-1><a href="JavaScript:getCoords()">GetCoords
- returns the coords</a></font></font>
<br> <font face="Arial,Helvetica"><font size=-1><a href="JavaScript:setFactor(60,30)">setFactor
- set speed up</a></font></font>
<br> <font face="Arial,Helvetica"><font size=-1><a href="JavaScript:setStartInc(-
1,0)">Left(Start)</a></font></font> <font face="Arial,Helvetica"><font size=-1><a
href="JavaScript:setStartInc(1,0)">Right(Start)</a></font></font>
</body>
</html>
```

Between the script tag you see a number of javascript functions. These are activate through ordinary html hyper links e.g.. :

```
<a href="JavaScript:getCoords()"GetCoords - returns the coords</a>
```

which when clicked activate the javascript function

```
function getCoords()
{
   xpos=document.scroller.getX();
   ypos=document.scroller.getY();
   x=document.scroller.getCursorX();
   y=document.scroller.getCursorY();
   alert(xpos + " : " + ypos + " : "+ x + " : " + y);
}
```

This function makes 4 calls to the applet named "scroller" which are located in the same html-document: getX(), getY(), getCursorX(), getCursorY(). In the end it show the result in a popup dialog.

In the scoller applet the following methods are included:

Method	Description
`setPos(int x, int y)`	Sets the position of the canvas x: The x-coordinate of top left position of the canvas to be drowned y: The y-coordinate top left position of the canvas to be drowned Returns no data
`setFactor(int whenThrowed, int autoRun)`	Sets the speed off the canvas `whenThrowed`: Speed when the canvas is thrown by the user (integer 1=slow... 30=fast) `autoRun`: speed when the canvas run in auto mode (integer 1=slow; 30=fast) Returns no data
`setStartInc(int x, int`	Set the direction when the canvas is running in auto mode in the x and y direction.

`y)`	x-direction: -1 = left-to-right; 1=right-to-left y-direction: -1 =buttom-to-top; 1=top-to-buttom Returns no data
`startThread()`	Start the thread running the component. Note that this is not the same as starting the canvas. If you want the canvas to start in auto mode you has to set the auto mode by calling the "setStartRun()"-method first with the parameter **true** and then call the startThread()-method. (Startthread() is like starting the motor in a car but don't put in gear. Nothing happens!) Returns no data
`stopThread()`	Stop the thread running the component. Returns no data
`setStartRun(boolean b)`	Sets the canvas in auto run mode if the parameter is true. If the parameter is false it stop the auto run mode. Returns no data
`int getX()`	Returns the x-coordinate of the canvas position as an integer
`int getY()`	Returns the y-coordinate of the canvas position as an integer
`int getCursorX()`	Returns the x-coordinate of the mouse position relative to the top left corner of the applet as an integer. Note that a value will be returned even though the cursor is outside the area of the applet.
`int getCursorY()`	Returns the x-coordinate of the mouse position relative to the top left corner of the applet as an integer. Note that a value will be returned even though the cursor is outside the area of the applet.

You can call javascript from the component if you include the netscape folder under the folder where you have stored the .class files. You also have to include a MAYSCRIPT command in the applet tag.

Example 2 and 3 - sound

You properly have heard about the difficulties play sound in the two major browsers: MS Microsoft and Netscape Navigator from the same code. This gets even worse when applets comes in play, because applets only supports .au files. By combining the strengths of Java and JavaScript and use some smaller html-tricks we are apple to overcome all obstacles.

Important: The sound in example2 and example3 will NOT play from a local hard disk in Netscape Navigator. You have to upload it to a internet server with at least one period in the domain name e.g.. nessal.dk to see the examples in Netscape. You cannot even use Microsoft Personal Webserver to simulate the net, because this server often use "localhost" as a servername without periods.

The trick is to use javascript to snooze whether the user has a MS Explore browser or a Netscape Navigator browser. According to this the appropriate code is written to the document. This is done in the "sound.html" in a separate frame, "sound". Sound.html is listed below:

```
<HTML>
<head><TITLE>Sound</TITLE>
<SCRIPT TYPE="text/javascript" LANGUAGE="JavaScript">
<!--// Hide JavaScript from Older Browsers

 var ver = navigator.appVersion;
 var sound="";

 function load()
 {
         sound=parent.scroller.getSound();
         if (ver.indexOf("MSIE") != -1)
         {
               document.open();
               document.writeln('<html><head><title></title></head><body bgcolor=#ffffc4>');
               document.writeln('<BGSOUND SRC="'+sound+'" LOOP="1">');
               document.writeln('</body></html>');
               document.close();
         }
         else
         {
               document.writeln('<EMBED AUTOSTART="true" LOOP="1" WIDTH="2" HEIGHT="0"
C="'+sound+'">');
         }
}
 // -->
</SCRIPT>
</HEAD>

<BODY BGCOLOR="#FFFFC4" onLoad="load()">

<!-- Thanks to http://camalott.com/html/sounds/javascript.html"> -->
</BODY>
</HTML>
```

When this html-document is loaded the javascript-function load() is activated. The first line gets the melody to be played:

```
sound=parent.scroller.getSound();
```

The reason for this construction is that the sound is a parameter in the scroller.html - document, which is located in the scroller frame.

According to the users browser the following line is activated:

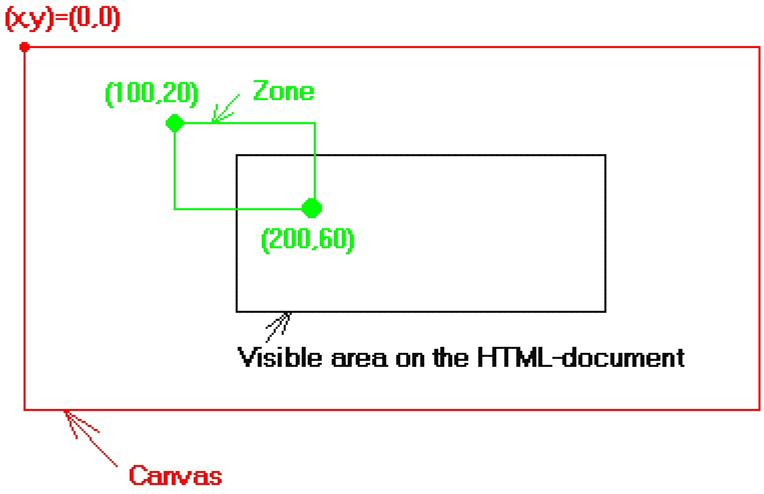
MS Explore: `<BGSOUND SRC="'+sound+'" LOOP="1"`

Netscape navigator: `<EMBED AUTOSTART="true" LOOP="1" WIDTH="2" HEIGHT="0" SRC="'+sound+'"`

Notice the `WIDTH="2" HEIGHT="0"` settings. This is "magic" settings in Netscape Navigator which hide the sound control from the user.

Example 4 - Zones

As with mouse over areas you can define areas where sudden action will be taken. These areas are called zones. This way you can control the canvas directly from the input file. You can of course control



the canvas from JavaScript, but in many situation it is more simpler just to use zones.

The reference point when defining a zone is the top left point of the canvas. When you define the zone you define the coords of the top-left and the botton-right of a rectangle. You can then specify some actions to be taken when the top-left point of the visible area of the area are in this zone:

4 = Direction information

Example:

```
4#0#40#100#60#-1#0
```

When the top-left point of the visible area is in the zone given by (x,y)=(40,100) and (x1,y1)=(100,60) the paremeter startincx and startincy (which are used to describe the direction of the canvas in autorun mode) is set to

startincx=-1
startincx=0

5 = Wait zone

Example:

```
5#0#0#100#10#5000#0
```

When the top-left point of the visible area is in the zone given by (x,y)=(0,0) and (x1,y1)=(100,10) the canvas will wait in 5000 ms (=5 seconds) before going on. You have to leave the zone before you can activate the waiting time again.

The last ‘0’ indicate that this should take action the first time the zone is entered. If this parameter is changes to ‘1’ you can enter the zone once without any waittime. When you leave the zone and come back you will have to wait.

6 = Jump zone

Example:

```
6#700#40#700#60#0#80
```

When the top-left point of the visible area is in the zone given by (x,y)=(700,40) and (x1,y1)=(700,60) the canvas the visible area will change so the top-left point of the visible area now is (0,80).

7 = New URL zone

Example:

```
7#700#40#700#60#http://www.nessal.dk#main
```

When the top-left point of the visible area is in the zone given by (x,y)=(700,40) and (x1,y1)=(700,60) the URL http://www.nessal.dk will be showned in the target frame “main”.

8 = drawRect zone

Example:

```
8#600#120#1200#400#000000
```

Draw a filled rectangle specified with the coords (x,y)=(600,120) and (x1,y1)=(1200,400) in black color. The color is specfied by the last parameter in hexidecimal RGB values.

Registrer

If you registrer the scroller component you will be emailed a file “license.txt” which should be placed in the same folder as the html-document containing the applet. If the folder is placed on the domain specified in your order the text “Not registred - ©nessal, Denmark" will be removed.