jQuery

mOover

Ultra Slick & Fancy Content Slider

Usage Documentation

For jQuery mOover v 1.5

By Vladimir Kharlampidi
@nolimits4web
The iDangero.us
http://www.idangero.us/

mOover v1.5
@idangerous
http://www.idangero.us/sliders/moover/

*Thank you for purchasing the **mOover**! If you have any questions that are beyond the scope of this help file, please please feel free to contact us via our support ticket form here.*

Table of Contents

1. mOover

1.1. Slider Integration (HTML and CSS)

It is very easy to integrate mOover into your page. So let's first of all look at the HTML code we need to use:

First of all you need to attach the jQuery library and mOover scripts to your document. Add the following code to the HEAD section or in the end of the BODY section:

```
<script src="path_to_jquery/jquery.js"></script>
<script src="path_to/moover/jquery.id.moover-1.3.min.js"></script>
```

We also need to link the Moover's CSS file

```
<link rel="stylesheet" href="path_to/moover/jquery.id.moover-1.3.css" type="text/
css" >
```

Then insert the following code inside of BODY:

```
<div id="moover">

  <!-- First Slide. Just text -->
  <div class="moover-slide">
    <!-- Container with text -->
    <div class="moover-text">
      <p>Text Line 1</p>
      <p>Text Line 2</p>
    </div>
  </div>

  <!-- Second Slide. Text and Image -->
  <div class="moover-slide">
    <!-- Back Image -->
    <img src="image.jpg" width="800"/>
    <!-- Container with text -->
    <div class="moover-text">
      <p>Text Line 1</p>
      <p>Text Line 2</p>
    </div>
  </div>

</div>
<!-- Pagination -->
<div class="moover-navigation"></div>

<!-- Control Elements, if Required -->
<a href="#" class="moover-next">Next Slide</a>
<a href="#" class="moover-prev">Previous Slide</a>
<a href="#" class="moover-play">Play</a>
<a href="#" class="moover-stop">Stop</a>
```

At the example above we used slider with a two slides. As you can see the code is pretty simple.

Slides:

- Every slide must have a "moover-slide" class
- Text block inside of slide must have a "moover-text" class
- Use only Paragraphs (<P>) inside of text block
- Slide's image must be the first child element inside of slide without additional wrapping elements
- **Do not forget to specify image width**

Pagination

If you want to use "pagination" add somewhere div with a "moover-navigation" class (for example). Pagination elements will be add inside of this div on slider's initialization.

mOover will automatically add there as many DIVs (with a "moover-switch" class) as many slides you have. Active elements will have "moover-active-switch" class. These classes can be useful for styling.

Control elements

mOover can be controlled by external buttons, you can add them if you need it. Note that CSS classes in the example (moover-next, moover-play, etc.) above must be specified in mOover parameters and may have other names (see next chapter)

Slider container

All divs with a "moover-slide" class wrapped with a one DIV with a "moover" id attribute. This is the main Slider container. We will use it for Slider initialization. You can stylize this element as you wish, wrap it with any additional elements you may need, but do not put anything inside except slides!

Now, let's look at the CSS code

We need only the last part of the jquery.id.moover-1.3.css file where we need to specify mOover's width and height:

```
/* Set mOover's width and height! */
.moover, .moover-slide {
	/* Do not forget to set your width and height */
	width:810px;
	height:320px;
}
```

If you will use pagination you will also need to add these rules with your styles:

```
.moover-switch {
	/* Your custom styles here */
}
.moover-active-switch {
	/* Your custom styles here */
}
```

That is all about required HTML and CSS code. As you can see it's very simple

1.2. Images Pre-Loader

mOover comes with in-built pre-loader of images which are used in slides. While images are loading the mOover's container has "moover-loading" class. So you can add a pre-loader image using css, for example:

```
.moover-loading {
	background: url(pat/to/loader.gif);
}
```

After all images will be loaded, mOover will start slideshow and 'moover-loading' class will be removed from container.

If you do not want to use in-build images pre-loader you can switch it off in parameters on moover's initialization (see next chapter).

1.3. Slider Initialization (Java Script)

We need to initialize the mOover after the document is ready (is loaded). Use the following formatting inside of <script> tag (if you use it inside of your document file)

```
jQuery(function(){
	$("#moover").moover()
})
```

That is done! Now mOover will start slideshow but with default parameters.

If you want to change effect or effect's settings you need to init it with parameters:

```
jQuery(function(){
	$("#moover").moover(parameters)
})
```

Now let's look at the list of available parameters:

Parameter	Type of variable	Example	Required	Default	Description
Images Preloader					
preLoader	*boolean*	`false`	optional	`true`	Set to 'false' if you want to disable in-built images pre-loader
Pagination / Controls					
navigation	*string*	`".moover-navigation"`	optional	`-`	CSS selector of the container with pagination

Parameter	Type of variable	Example	Required	Default	Description
navigationActive	*boolean*	`true`	optional	`false`	Set to "true" if you want to make pagination buttons 'clickable'. In this case clicking on some pagination button will cause transition to appropriate slide
nextButton	*string*	`'a.moover-next'`	optional	-	CSS selector of the next button, which will cause transition to next slide when the mOover is stopped
prevButton	*string*	`'a.moover-prev'`	optional	-	CSS selector of the previous button, which will cause transition to previous slide after animation of the currently active slide
stopButton	*string*	`'a.moover-stop'`	optional	-	CSS selector of the "stop" button, which will stop auto play after animation of the currently active slide

Parameter	Type of variable	Example	Required	Default	Description
playButton	*string*	`'a.moover-play'`	optional	-	CSS selector of the "play" button, which will enable auto play if the mOover is stopped
Text Effect					
effect	*string*	`"slide"` or `"typewriter"` or `"pusher"`	optional	`"slide"`	Text Effect
Image Effects					
moveImage	*boolean*	`true`	optional	`false`	if true, than image will be move during the selected text effect
scaleImage	*number*	`1.2`	optional	`1`	Scale multiplier for the image. It will be scaled to selected value during the text effect
moveOutImage	*boolean*	`true`	optional	`true`	If true, then the image will be move-out in the end of transition
moveWidth	*number*	`200`	optional	`100`	Image will be moved during the text effect on this value/2 (in px) This value is also used by "slide" text effect. Text will be moved on this value

Parameter	Type of variable	Example	Required	Default	Description
slideTime	*number*	`500`	optional	`1000`	Image of new slide will slide-in for the time equal to this value (in ms). This value is also used by the "slide" text effect. Text lines of new slide will slide-in for this time.
moveTime	*number*	`3600`	optional	`3000`	During the text effect image will move and scale (if moveWidth and scaleImage are specified) for the time equal to this value (in ms) This value is also used by "slide" text effect. Text lines will slowly move in the middle of effect for this time
Call Back Functions					
onStart	*function*	`function() {alert("Hello")}`	optional	-	custom function in the beginning of slideshow
onSlideSwitch	*function*	`function() {alert("Hello")}`	optional	-	custom function will be executed in the beginning of transition to next slide

Parameter	Type of variable	Example	Required	Default	Description
onSlideSwitchEnd	*function*	`function() {alert("Hello")}`	optional	-	custom function will be executed right after transition to next slide
Manual Mode					
manualMode	*boolean*	`true`	optional	`false`	Set to 'true' to enable "Manual Mode". In Manual mode slider will be stopped after animation of each slide. It's like to disable auto play. In this case navigation through slides will be only available via active pagination or control buttons
Debugging					
disableCSS3	*boolean*	`false`	optional	`false`	disable or enable CSS3 effects. If your transitions do not use rotation, translates and scale effects, it is better to disable it (set to "true")

Let's see not example of initialization with parameters:

```
jQuery(function(){
      $("#moover").moover({
            /* Pagination */
            navigation : ".moover-navigation",

            /* Effect */
            effect : "typewriter",

            /* Image Effects */
```

```
            moveImage : true,
            scaleImage : 1.2,
            moveWidth : 200,
            moveOutImage : true,

            /* Image Durations */
            moveTime : 4000,
            slideTime : 1000,

            /* Callbacks */
            onStart : function() {
                  // Do something
            },
            onSlideSwitch : function() {
                  // Do something
            }
      })
})
```

Now let's learn more about effects and its parameters:

1.3. Effects

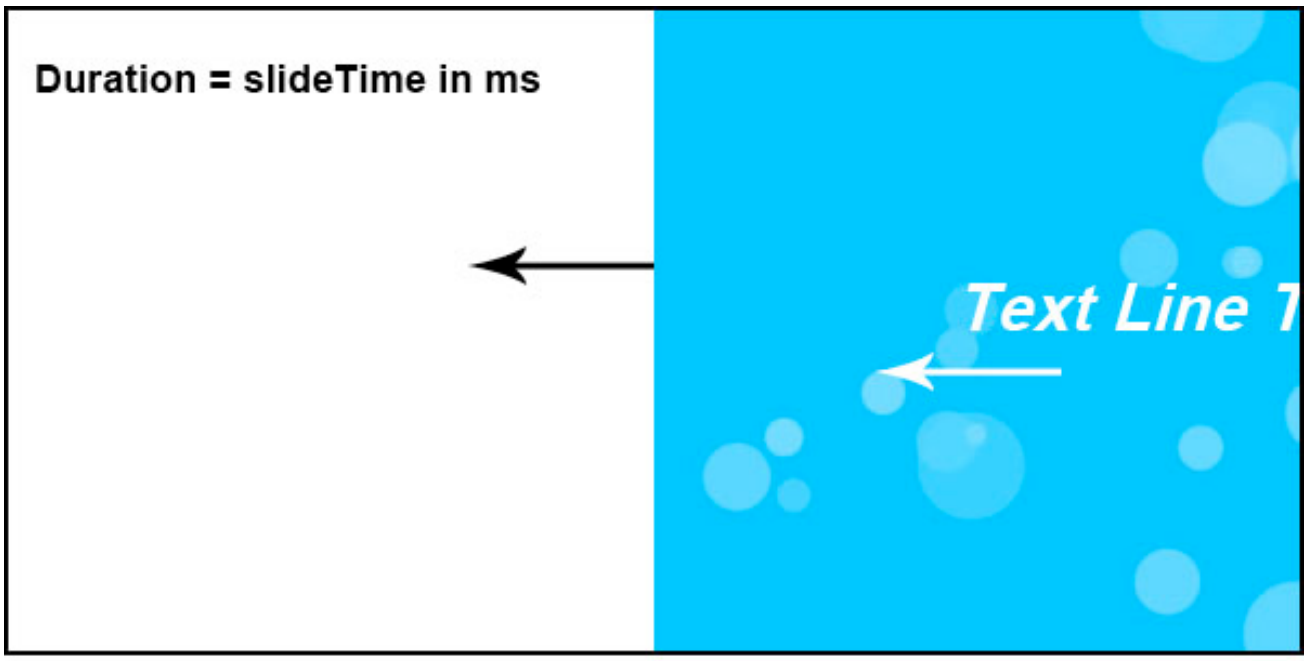
As mentioned above there are 3 text effect:

Slide Effect

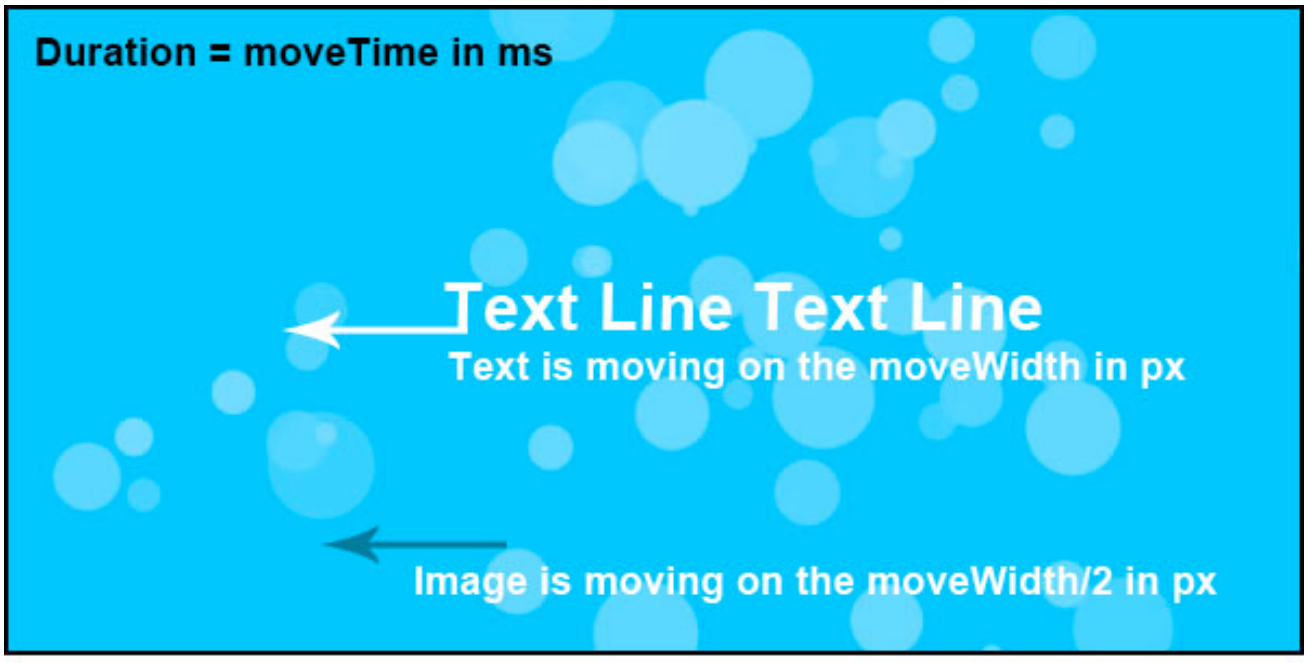
In this effect text moves inside of slide line-by-line. This effect consist of three stages:

- **Stage 1**. Image and text lines **slide-in** for the time equal to **slideTime** parameter. During this, text will be transformed from **initTextTransform** to **middleTextTransform** values.
- **Stage 2**. Image (if moveImage and scaleImage are specified) and text lines are moving during the time equal to **moveTime** parameter. Text moves on the width equal to **moveWidth**, image moves on the width equal to **moveWidth/2** .
- **Stage 3**. Image (if moveOutImage = true) and text lines **slide-out** for the time equal to **slideTime** parameter. During this, text will be transformed from **middleTextTransform** to **finalTextTransform** values.

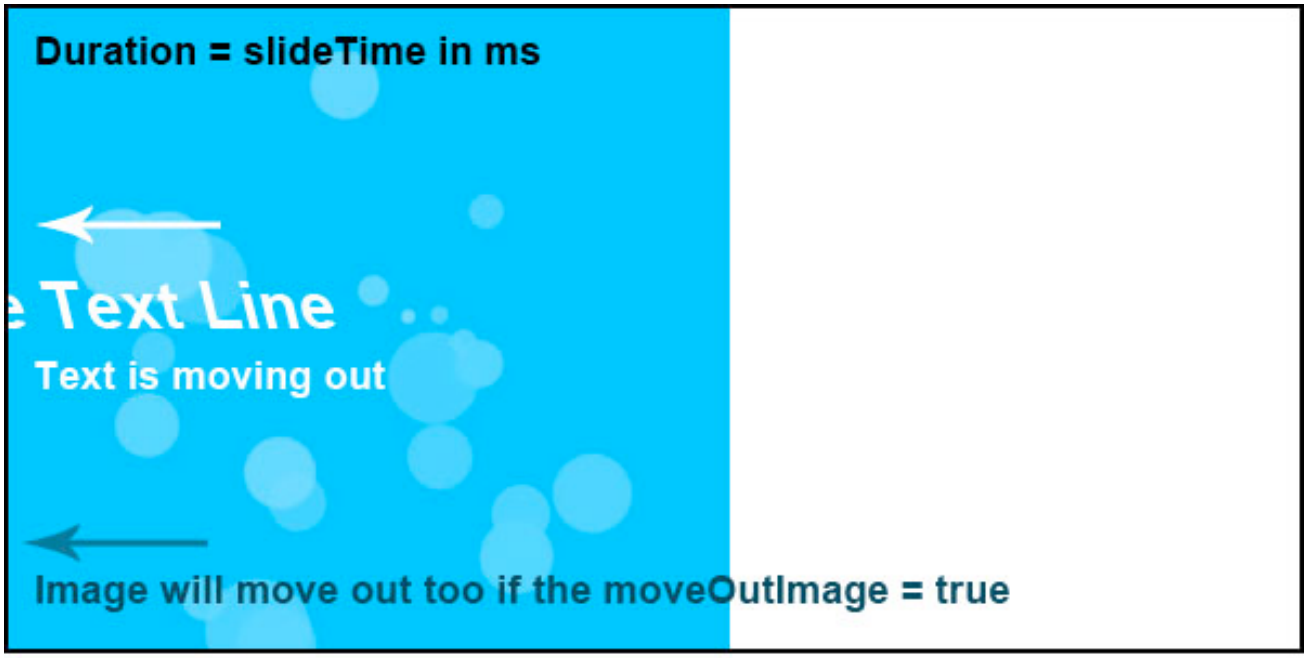
Stage 1



Stage 2



Stage 3

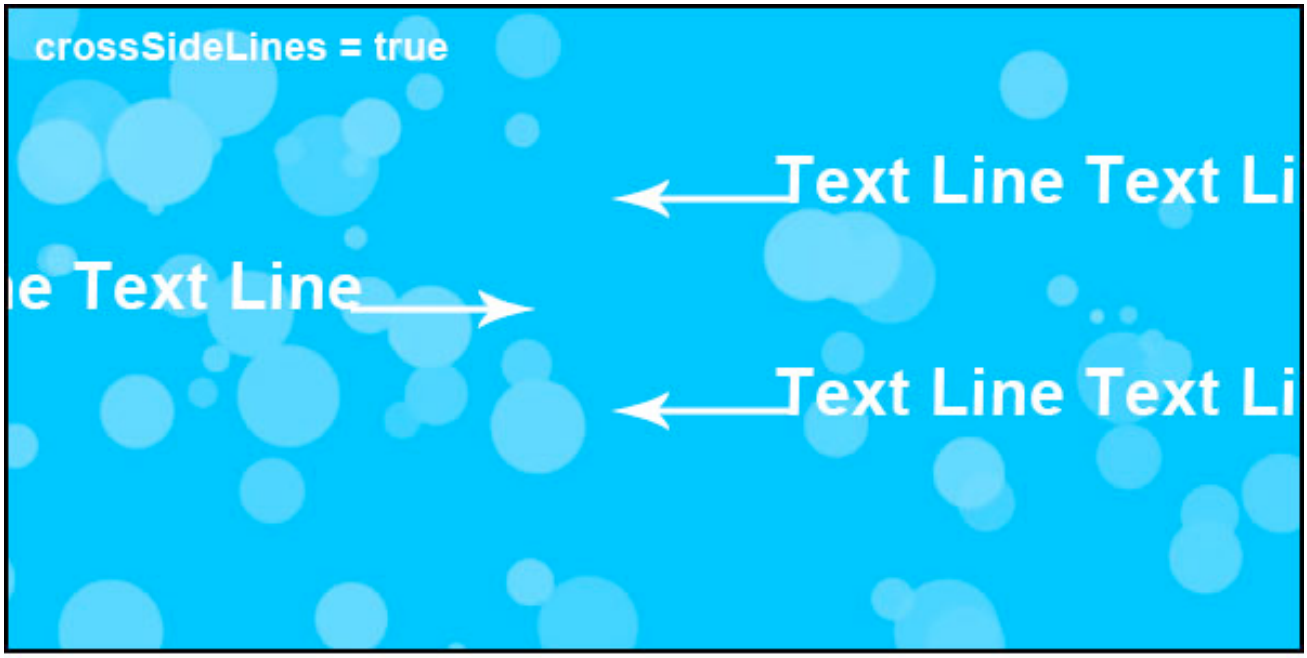


Slider Container

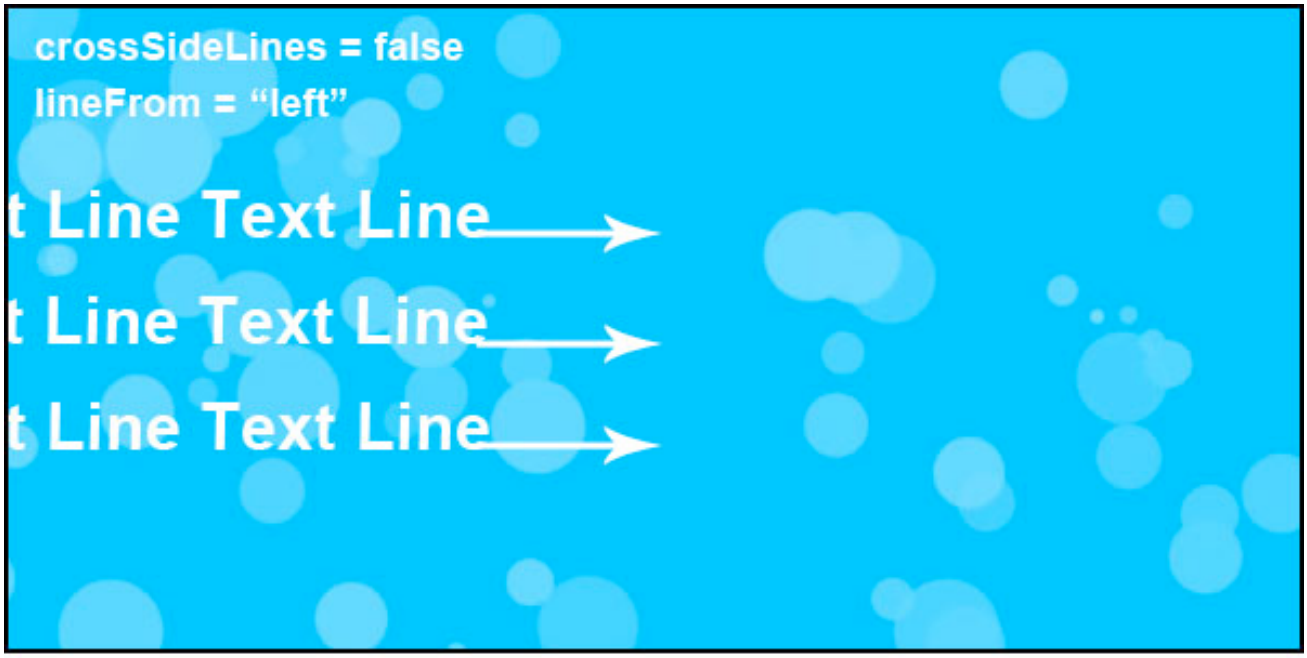
Let's look at the parameters of "slide" effect:

Parameter	Type of variable	Example	Required	Default	Description
textLineDelay	*number*	`100`	optional	`0`	Delay in ms between text lines
crossSideLines	*boolean*	`false`	optional	`true`	if true then every next line will slide from different side
lineFrom	*string*	`"left"`	optional	`"right"`	If "right" then first line will slide-in from the right side
initTextTransform	*string*	`"rotate(90deg)"`	optional	`"scale(1,0) skewX(40deg)"`	Initial transformation of text before slide-in . Use CSS3 transform syntax

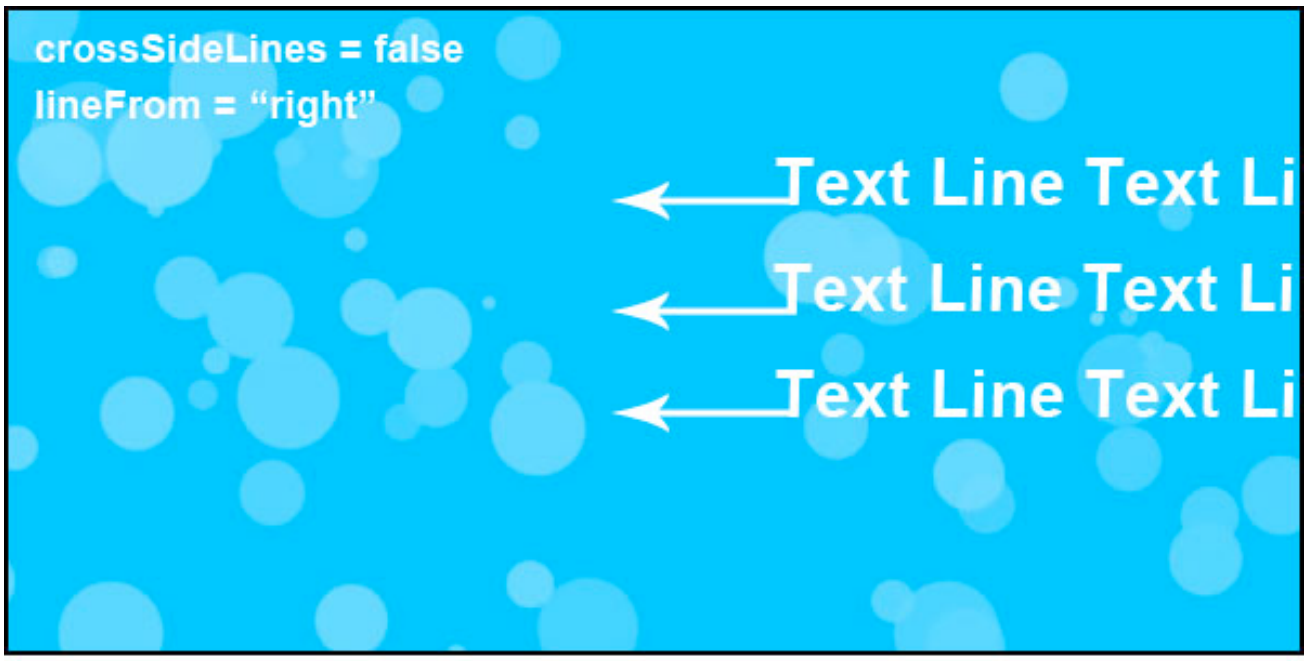
Parameter	Type of variable	Example	Required	Default	Description
middleTextTransform	*string*	`"rotate(0deg)"`	optional	`"scale(1,1) skewX(0deg)"`	Middle transformation of text on "stage 2" . Use CSS3 transform syntax
finalTextTransform	*string*	`"rotate(-90deg)"`	optional	`"scale(1,0) skewX(-40deg)"`	Final transformation of text on slide-out . Use CSS3 transform syntax
afterSlideHoldTime	*number*	`3000`	optional	`0`	After all lines slide in, mOover will hold them this time (in ms) until the "stage3"



Slider Container



Slider Container



Slider Container

Here is the example of Slider initialization with "slide" parameters:

```
jQuery(function(){
      $("#moover").moover({
            /* Pagination */
            navigation : ".moover-navigation",

            /* Effect */
            effect : "slide",

            /* Image Effects */
            moveImage : true,
            scaleImage : 1.2,
```

```
            moveWidth : 200,
            moveOutImage : true,

            /* Image Durations */
            moveTime : 4000,
            slideTime : 1000,

            /* Slide settings */

            crossSideLines : false,
            lineFrom : "left",
            textLineDelay : 100,
            initTextTransform : "rotate(90deg)",
            middleTextTransform : "rotate(0deg)",
            finalTextTransform : "rotate(-90deg)",


            /* Callbacks */
            onStart : function() {
                  // Do something
            },
            onSlideSwitch : function() {
                  // Do something
            }
      })
})
```

Important Notes:

- do not need to specify all effect parameters. Use only those that you want to change from default ones.
- to learn more **transform syntax** please visit this site: http://www.w3.org/TR/css3-2d-transforms/#transform-functions
- if you are using custom transforms, not default ones, you need to specify all three transform parameters with the same syntax. For example transform from "scale(1) rotate(90deg)" to "rotate(30deg)" will not work! Use from "scale(1) rotate(90deg)" to "scale(1) rotate(30deg)".

Pusher Effect

In this effect text lines fall down, and every previous line pushes the line after it. This effect is also consist of three stages

- **Stage 1**. Only image **slides-in** for the time equal to **slideTime** parameter.
- **Stage 2**. Image moves (*if moveImage = true*) and scales (*if scaleImage not equal to 1*) during the time equal to **moveTime** parameter. Text lines fall down one by one.
- **Stage 3**. Image **slides-out** for the time equal to **slideTime** parameter (*if moveOutImage = true*). Text container fall down outside the slider container for the time equal to the **slideTime**.

Pusher parameters:

Parameter	Type of variable	Example	Required	Default	Description
pushTime	*number*	200	optional	100	Time in ms of line falling animation
pushDelay	*number*	200	optional	300	Delay in ms between text lines
afterPushHoldTime	*number*	1000	optional	1500	After all lines fall down on the center, mOover will hold them this time (in ms) until the "stage3"

Here is the example of Slider initialization with "pusher" parameters:

```
jQuery(function(){
	$("#moover").moover({
		/* Pagination */
		navigation : ".moover-navigation",

		/* Effect */
		effect : "pusher",

		/* Image Effects */
		moveImage : true,
		scaleImage : 1.2,
		moveWidth : 200,
		moveOutImage : true,

		/* Image Durations */
		moveTime : 4000,
		slideTime : 1000,

		/* Pusher settings */

		pushTime : 350,
		pushDelay : 600,
		afterPushHoldTime : 3000,

		/* Callbacks */
		onStart : function() {
			// Do something
		},
		onSlideSwitch : function() {
			// Do something
		}
	})
})
```

It was pretty simple and again do not need to specify all effect parameters. Use only those that you want to change from default ones.

Typewriter Effect

Typewriter is the most powerful and eye-catching effect of mOover. It is intended to "print" text chars, but also allows to print any other HTML elements like images.

Typewriter Effect consist of three stages:

- **Stage 1**. Only image **slides-in** for the time equal to **slideTime** parameter.
- **Stage 2**. Image moves (*if moveImage = true*) and scales (*if scaleImage not equal to 1*) during the time equal to **moveTime** parameter. And mOover start to print text character by character.
- **Stage 3**. Image **slides-out** for the time equal to **slideTime** parameter (*if moveOutImage = true*). Text container fades out for the time equal to **slideTime**.

Effect parameters:

Parameter	Type of variable	Example	Required	Default	Description
charTime	*number*	`200`	optional	`100`	Every character will be "printed" for this time in ms
charDelay	*number*	`200`	optional	`40`	Delay in ms between characters
textHoldTime	*number*	`1000`	optional	`1500`	After all characters are printed, mOover will hold them this time (in ms) until the "stage3"
waitForImage	*boolean*	`false`	optional	`true`	If false, then first and second stages will start at the same moment. mOover will not wait until the image slides-in, it will print the text immediately

Parameter	Type of variable	Example	Required	Default	Description
initCharTransform	*string*		optional	`scale(0,0) rotate(0deg)`	Initial transformation of the character
middleCharTransform	*string*		optional	`scale(1.2,1.3) rotate(5deg)`	middle transformation of the character
finalCharTransform	*string*		optional	`scale(1,1) rotate(0deg)`	final transformation of the character
charOrigin	*string*	`"left top"`	optional	`"center bottom"`	CSS3 transform-origin property of the character
charEase	*string*	`"ease-in"` or `"ease-out"` or `"ease"` or `"ease-in-out"`	optional	`"ease-out"`	Ease function of the character animation

Character Transforms

First of all every characters will appear with an **initCharTransform** transformation, then it will be transformed to **middleCharTransform** for the time equal to **charTime**, and after it will be transformed to **finalCharTransform** for the **charTime** ms.

Important Notes:

- to learn more **rotate, scale, translate** values and how do they work please visit this site: http://www.w3.org/TR/css3-2d-transforms/#transform-functions
- to learn how the **origin** works look here: http://www.w3.org/TR/css3-2d-transforms/#transform-origin-property
- to learn how the **ease** works and what values it can accept look here: http://dev.w3.org/csswg/css3-transitions/#transition-timing-function_tag
- if you are using custom transforms, not default ones, you need to specify all three transform parameters with the same syntax. For example transform from "scale(1) rotate(90deg)" to "rotate(30deg)" will not work! Use from "scale(1) rotate(90deg)" to "scale(1) rotate(30deg)".

Here is the example of Slider initialization with "typewriter" parameters:

```
jQuery(function(){
      $("#moover").moover({
            /* Pagination */
```

```
            navigation : ".moover-navigation",

            /* Effect */
            effect : "typewriter",

            /* Image Effects */
            moveImage : true,
            scaleImage : 1.2,
            moveWidth : 200,
            moveOutImage : true,

            /* Image Durations */
            moveTime : 4000,
            slideTime : 1000,

            /* TypeWriter settings */

            waitForImage : false,
            charDelay : 60,
            charTime : 300,
            textHoldTime : 3000,

            /* Callbacks */
            onStart : function() {
                  // Do something
            },
            onSlideSwitch : function() {
                  // Do something
            }
      })
})
```

Typewriter allows to print not only text characters, but also any HTML elements, like images or entire words. If you want to print not only character-by-character, but some word or image or etc, wrap these elements with span:

```
<!-- First Slide will print text char by char -->
<div class="moover-slide">
      <div class="moover-text">
            <p>First slide text line 1</p>
            <p>Second slide text line 2</p>
      </div>
</div>
<!-- Second slide will print chars, words and elements -->
<div class="moover-slide">
      <div class="moover-text">
            <p>Phrase 1. <span>Word</span> <span>by</span> <span>word</span></p>
            <p>This will print image: <span><img src="myimage.png" /></span></p>
      </div>
</div>
```

In the example above text of first slide will be printed char-by-char.

But let's look at the second slide - all elements wrapped with SPAN will be printed like one char. In the first line phrase "Phrase 1." will be printed char-by-char, but the next phrase "Word by word" will be printed word-by-word because every word is wrapped with span. In the second line "this will print image" - char-by-char, and then image will be printed. So here are couple of important notes:

1. If you want to keep some element intact - wrap it with SPAN tag.
2. Everything except the TEXT and SPANs will be ignored by typewriter and will not be outputted. For example
   Phrase 1, some other text and image :
   Image will be ignored and will not be outputted! Use spans:
   Phrase 1, some other text and image :
3. The same with inline elements like <a>, and - wrap them with SPANs

Fader Effect

In this effect text lines fade in from different directions. This effect is also consist of three stages.

This effect might be used only with mOover-absolute (read bellow). You'll need to set left and top position for every paragraph in slide.

- **Stage 1**. Only image **slides-in** for the time equal to **slideTime** parameter.
- **Stage 2**. Image moves (*if moveImage = true*) and scales (*if scaleImage not equal to 1*) during the time equal to **moveTime** parameter. Text lines fade in one by another.
- **Stage 3**. Image **slides-out** for the time equal to **slideTime** parameter (*if moveOutImage = true*). Text container fades out for the time equal to the **slideTime**.

Fader parameters:

Parameter	Type of variable	Example	Required	Default	Description
fadeTime	*number*	`200`	optional	`500`	Time in ms of line fading animation
fadeDelay	*number*	`200`	optional	`300`	Delay in ms between text lines
fadeType	*string*	`"vertical"`	optional	`"horizontal"`	Type of fade animation. Could be vertical or horizontal
afterFadeHoldTime	*number*	`1000`	optional	`3000`	After all lines fade in, mOover will hold them this time (in ms) until the "stage3"

Here is the example of Slider initialization with "fader" parameters:

```
jQuery(function(){
      $("#moover").moover({
            /* Pagination */
```

```
            navigation : ".moover-navigation",

            /* Effect */
            effect : "fader",

            /* Image Effects */
            moveImage : true,
            scaleImage : 1.2,
            moveWidth : 200,
            moveOutImage : true,

            /* Image Durations */
            moveTime : 4000,
            slideTime : 1000,

            /* Pusher settings */

            fadeTime : 300,
            fadeDelay : 200,
            fadeType : "vertical",
            afterFadeHoldTime : 2000,

            /* Callbacks */
            onStart : function() {
                  // Do something
            },
            onSlideSwitch : function() {
                  // Do something
            }
      })
})
```

It was pretty simple and again do not need to specify all effect parameters. Use only those that you want to change from default ones.

mOover-Absolute

Using mOover-Absolute you can set slide's elements in any place, by setting position:absolute to paragraphs. For this case you need to add special class for the slide:

```
<div class="moover-slide moover-absolute">
      <div class="moover-text">
            <p class="line">Text Line 1</p>
            <p class="logo"><span><img src="slide1.jpg" /></span></p>
      </div>
</div>
```

Now you can specify position of this "absolute" paragraphs with CSS, relatively to the slider container:

```
.line1 {
      position:absolute;
      left:100px;
      top:100px;
}
.logo {
```

```
	position:absolute;
	right:20px;
	bottom:200px;
}
```

1.4. Every Slide - New Story!

mOover allows to specify totally different effects for the different slides, for this case you will need to add one more parameter "effects" and specify effects for different slides:

```
jQuery(function(){
	$("#moover").moover({
		/* First of all default mOover paramters */
		navigation : ".moover-navigation",

		effect : "slide",

		moveImage : true,
		scaleImage : 1.2,
		moveWidth : 200,
		moveOutImage : true,

		moveTime : 4000,
		slideTime : 1000,


		waitForImage : false,
		charDelay : 60,
		charTime : 300,
		textHoldTime : 3000,

		/* Now we can change some parameters for the required slides
		or totally change effect: */

		effects : {

			/* Change effect for 1st slide */
			"1" : {
				effect : "slide",
				moveTime : 5000
			},

			/* Change effect for the 3rd slide */
			"3" : {
				effect : "pusher",
				moveTime : 5000,
				pushTime : 200,
				pushDelay : 500,
				moveOutImage : false,
				scaleImage : 0.7
			}
		}
	})
})
```

2. Transform and Timing Presets

From version 1.4 mOover accept so called 'transform' presets for 'slider' and 'typewriter' effects. These are already predefined and ready to use groups of the transform parameters :

- **initCharTransform, middleCharTransform, finalCharTransform** - for typewriter effect
- **initTextTransform, middleTextTransform, finalTextTransform** - for slide effect

There are following transform preset are available for Slide effect:
- 'none'
- 'skew'
- 'flippers-h'
- 'flippers-v'
- 'scale-full'
- 'scale-y'
- 'scale-x'
- 'rotation'
- 'hurricane'

For Typewriter effect:
- 'default'
- 'rotators'
- 'flippers-h'
- 'flippers-v'
- 'fall'
- 'fall-elastic'
- 'fall-reverse'
- 'rain-elastic'
- 'hurricane'

For typewriter there are also available 'timing presets'. Timing preset is a group of predefined "charTime" and "charDelay" options. Timing preset can accept the following values:
- 'very-fast'
- 'fast'
- 'medium'
- 'slow-short' - slow animation speed with short delay between characters
- 'slow'
- 'slow-long' - slow animation speed with long delay between characters

- 'very-slow'

Example of usage:

```
jQuery(document).ready(function(){
	$("#moover").moover({
		effect : "slide",
		transformPreset : 'hurricane'
	})
/*---- OR ----*/
	$("#moover").moover({
		effect : "typewriter",
		transformPreset : 'rotators',
		timingPreset : 'fast'
	})
})
```

If using different effects for different slides:

```
jQuery(document).ready(function(){
	$("#moover").moover({
		effects : {
			'1' : {
				effect:'slide',
				transformPreset : 'hurricane'
			},
			'2' : {
				effect:'typewriter',
				transformPreset : 'fall'
			},
			//.....
		}

	})
})
```

Why to use presets? It is good to use them if you don't familiar with CSS3 transform syntax and want to save some time while configuring mOover.

Example without preset:

```
jQuery(document).ready(function(){
	$("#moover").moover({
		effect : "typewriter",
		initCharTransform : "scale(1,1) translateY(-1000px)",
		middleCharTransform : "scale(1,0.5) translateY(0px)" ,
		finalCharTransform : "scale(1,1) translateY(0px)",
		charOrigin : 'center bottom',
		charTime:100,
		charDelay:40
	})
})
```

or with 'rotators' preset:

```
jQuery(document).ready(function(){
	$("#moover").moover({
		effect : "typewriter",
```

```
            transformPreset : "rotators",
            timingPreset : "fast"
        })
})
```

3. Using Multiple mOovers

You can use as many instances of mOover as you may need on one page. Let's see on the example:

```
<!-- First mOover -->
<div class="moover moover1">
  <div class="moover-slide"> <img src="i/slides/slide3.jpg" width="900" />
    <div class="moover-text">
      <p>There are 3 different moovers</p>
      <p>with 3 different effects</p>
    </div>
  </div>
  <div class="moover-slide"> <img src="i/slides/slide2.jpg" width="900" />
    <div class="moover-text">
      <p>There are 3 different moovers</p>
      <p>with 3 different effects</p>
    </div>
  </div>
</div>
<!-- Second mOover -->
<div class="moover moover2">
  <div class="moover-slide"> <img src="i/slides/slide4.jpg" width="1010" />
    <div class="moover-text">
      <p>There are 3 different moovers</p>
      <p>with 3 different effects</p>
    </div>
  </div>
  <div class="moover-slide"> <img src="i/slides/slide5.jpg" width="1010" />
    <div class="moover-text">
      <p>There are 3 different moovers</p>
      <p>with 3 different effects</p>
    </div>
  </div>
</div>
<!-- Third mOover -->
<div class="moover moover3">
  <div class="moover-slide"> <img src="i/slides/slide2.jpg" width="900"/>
    <div class="moover-text">
      <p>There are 3</p>
      <p> different</p>
      <p>mOovers</p>
      <p>with 3 different</p>
      <p>effects</p>
    </div>
  </div>
  <div class="moover-slide"> <img src="i/slides/slide3.jpg" width="900"/>
    <div class="moover-text">
      <p>There are 3 different</p>
      <p>mOovers</p>
      <p>with 3 different</p>
      <p>effects</p>
    </div>
```

```
  </div>
</div>

jQuery(function(){
      $(".moover1").moover(moover1Parameters)
      $(".moover2").moover(moover2Parameters)
      $(".moover3").moover(moover3Parameters)
})
```

As you can see it was very simple. Only note that you probably may need to change CSS styles for these mOovers in the mOover's CSS file.

4. mOover's External API

After you init the mOover, it returns special object which is contain a lot of useful methods.

```
jQuery(function(){
      var moover = $(".moover").moover(parameters)
})
```

And now, "`moover`" (it may have other name) variable contains special object. Here its structure:

4.1 moover.redefine()

This function allows you to redefine default slider's parameters that were set on slider's initialization. It accepts the same parameters specified in table above. **But do not need to use all parameters, use only those you want to redefine!**

Here are some examples of its usage:

```
      //Change effect to "typewriter"
      $("#change-effect").click(function(){
            moover.redefine({
                  effect: "typewriter"
            })
      });


      //Disable CSS3 effects
      $("#no-css3").click(function(){
            moover.redefine({
                  disableCSS3:true
            })
      })
```

4.2 moover.hasCSS3()

This function is very useful to determine is the browser supports CSS3 transitions or not.

```
      if( moover.hasCSS3() ) {
            // Do something
            alert ("Hola! Your browser supports CSS3 transitions!");
      }
      else {
```

```
        // Do something
        alert ("Oops! Your browser do not support CSS3 transitions!");
}
```

4.3 moover.parameters

It is an object, contains current parameters of mOover, that you can request in any moment:

```
// Get the current effect
alert (moover.parameters.effect)

//Get the current moveWidth
alert (moover.parameters.moveWidth)
```

4.4 moover.stop()

Stop mOover after animation of currently active slide. Can be useful to create custom navigation buttons:

```
$('.my-stop-button').click(function(){
      moover.stop()
})
```

4.5 moover.play()

Launch auto play if mOover is stopped. Can be useful to create custom navigation buttons:

```
$('.my-play-button').click(function(){
      moover.play()
})
```

4.6 moover.next()

Launch transition to next slide (if mOover is stopped). Can be useful to create custom navigation buttons:

```
$('.my-next-button').click(function(){
      moover.next()
})
```

4.7 moover.prev()

Launch transition to previous slide after animation of currently active slide, or immediately if mOover is stopped . Can be useful to create custom navigation buttons:

```
$('.my-prev-button').click(function(){
      moover.prev()
})
```

4.8 moover.isStopped

Variable, it returns 'true' if the mOover is stopped:

```
if (moover.isStopped) {
      alert ('Slider is stopped, click on the Play button')
}
```

4.9 moover.manualMode

Set it to 'true' to enable "manual mode":

```
$('.enable-manual-mode').click(function(){
      moover.manualMode = true
}
```

To disable "manual mode" and enable auto play use this trick:

```
$('.disable-manual-mode').click(function(){
      moover.manualMode = false;
      moover.play();
})
```

5. $(element).mvTransform() additional jQuery method

This is additional cross-browser (not IE) jQuery method allows to apply CSS3 transforms (or CSS3 animation) to any element.

mvTransform(parameters)

Parameter	Type of variable	Example	Required	Default	Description
transform	*string*	`"rotate(90deg)"`	**required**	–	CSS3 transform property
time	*number*	`600`	optional	`0`	duration of animation
ease	*string*	`"ease-in"`	optional	`"ease"`	Transition easing function
origin	*string*		optional	`"50% 50%"`	Transform origin property

Parameter	Type of variable	Example	Required	Default	Description
delay	*number*	300	optional	0	Delay before transition (animation) will start

Here is example:

```
//Element will be transformed for 600ms after the 200ms delay
//with custom origin and ease

$("#somelement").click(function(){
      $(this).mvTransform(
            {
                  transform:"rotate(45deg) scale(0.5) translateX(200px)",
                  delay:200,
                  time:600,
                  ease:"ease-in-out",
                  origin:"0px 0px"
            }
      )
})
```

Please note! If you will set "**disableCSS3"** property on mOover's initialization to "**true**", then the mvTransform() function **will not work**!