[**Introduction**](https://scotch.io/tutorials/off-canvas-menus-with-css3-transitions-and-transforms#introduction)

Off Canvas Menus are used primarily with Mobile and touch devices and can provide an extremely useful and beautiful experience for your users. This tutorial will cover an introduction on them, a getting started guide, some improvement tricks, and then finally some awesome demos for you to build off of. Instead of JavaScript, we’ll use CSS3 Transitions and Translations for the animation making them smoother, better performing, and easier to tweak and customize.

What the heck are **Off Canvas Menus** anyways? I’m not really sure where this name or terminology originated, but if you haven’t heard of it before, they’re basically your mobile “side menus”. They live outside of the viewport and are only shown when a certain event occurs (which is usually a click toggling the menu). This has the obvious and similar benefit of dropdowns menus with a lot more flair.

**Here’s a basic demo:**

* [Home](https://scotch.io/)
* [Tutorials](https://scotch.io/tutorials)
* [Bar Talk](https://scotch.io/bar-talk)
* [Quick Tips](https://scotch.io/quick-tips)
* [Projects](https://scotch.io/projects)
* [About](https://scotch.io/about)



**SCOTCHY SCOTCH SCOTCH**

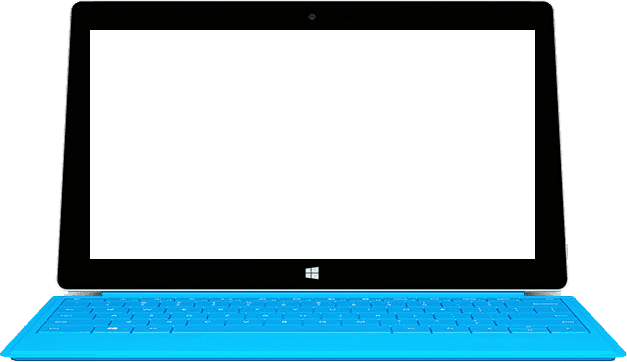
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus dignissim diam at dolor commodo posuere. Suspendisse aliquam lacus lectus, et bibendum mauris sollicitudin et. Integer volutpat felis nec urna lacinia ullamcorper. Pellentesque sem nisi, tincidunt in vestibulum et, pharetra fermentum risus. Sed ultricies molestie massa id hendrerit. Donec luctus luctus velit, tincidunt laoreet lorem dictum non. Etiam gravida sed lacus eu mollis. Nunc eu augue quis enim vestibulum mollis. Donec neque augue, aliquet sed fringilla a, convallis sit amet tortor.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus dignissim diam at dolor commodo posuere. Suspendisse aliquam lacus lectus, et bibendum mauris sollicitudin et. Integer volutpat felis nec urna lacinia ullamcorper. Pellentesque sem nisi, tincidunt in vestibulum et, pharetra fermentum risus. Sed ultricies molestie massa id hendrerit. Donec luctus luctus velit, tincidunt laoreet lorem dictum non. Etiam gravida sed lacus eu mollis. Nunc eu augue quis enim vestibulum mollis. Donec neque augue, aliquet sed fringilla a, convallis sit amet tortor.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus dignissim diam at dolor commodo posuere. Suspendisse aliquam lacus lectus, et bibendum mauris sollicitudin et. Integer volutpat felis nec urna lacinia ullamcorper. Pellentesque sem nisi, tincidunt in vestibulum et, pharetra fermentum risus. Sed ultricies molestie massa id hendrerit. Donec luctus luctus velit, tincidunt laoreet lorem dictum non. Etiam gravida sed lacus eu mollis. Nunc eu augue quis enim vestibulum mollis. Donec neque augue, aliquet sed fringilla a, convallis sit amet tortor.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus dignissim diam at dolor commodo posuere. Suspendisse aliquam lacus lectus, et bibendum mauris sollicitudin et. Integer volutpat felis nec urna lacinia ullamcorper. Pellentesque sem nisi, tincidunt in vestibulum et, pharetra fermentum risus. Sed ultricies molestie massa id hendrerit. Donec luctus luctus velit, tincidunt laoreet lorem dictum non. Etiam gravida sed lacus eu mollis. Nunc eu augue quis enim vestibulum mollis. Donec neque augue, aliquet sed fringilla a, convallis sit amet tortor.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus dignissim diam at dolor commodo posuere. Suspendisse aliquam lacus lectus, et bibendum mauris sollicitudin et. Integer volutpat felis nec urna lacinia ullamcorper. Pellentesque sem nisi, tincidunt in vestibulum et, pharetra fermentum risus. Sed ultricies molestie massa id hendrerit. Donec luctus luctus velit, tincidunt laoreet lorem dictum non. Etiam gravida sed lacus eu mollis. Nunc eu augue quis enim vestibulum mollis. Donec neque augue, aliquet sed fringilla a, convallis sit amet tortor.



**Some other resources and demos:**

* [Bootstrap 3 Off Canvas Menu](http://getbootstrap.com/examples/offcanvas/)
* [Foundation Off Canvas Menu](http://foundation.zurb.com/docs/components/offcanvas.html)

[**Getting Started**](https://scotch.io/tutorials/off-canvas-menus-with-css3-transitions-and-transforms#getting-started)

As with anything programming related, there are multiple ways or methods to the same solution. The solution that I’m going to show you is one that I like to use because it’s **simple**, extremely **obvious** to use, and will leave the door wide-open for you to **improve** and**tweak**. You’ll find after following these steps that you’ll be able to get much more creative with it than where the tutorial goes using only CSS.

**Heads Up!** This example doesn’t completely cover cross-browser support and is only meant to be the simplest bare-boned example possible. Read this[section](https://scotch.io/tutorials/off-canvas-menus-with-css3-transitions-and-transforms#browser-support-help) for that. Otherwise, you’ll have to use Chrome if you follow the steps step-by-step to get it working.

[**How it works**](https://scotch.io/tutorials/off-canvas-menus-with-css3-transitions-and-transforms#how-it-works)

For the purpose of this article and keeping naming convention consistent amongst already existing resources, we’ll continue to use the term *Canvas*. Although I think this is confusing because I feel that namespace is already reserved for [HTML5 Canvas](https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Canvas_tutorial). This has **absolutely nothing** to do with that. A name that would make more sense would be *Off Viewport Menus*.

Anyways, here’s the terminology broken down for the tutorial and a blueprint of how to make Off Canvas Menus:

* Wrapper
* Canvas
* Menu

[**The HTML**](https://scotch.io/tutorials/off-canvas-menus-with-css3-transitions-and-transforms#the-html)

From the blueprint above, lets build out the HTML. For this example, we’ll be assuming this a full-window Off Canvas Menu. So, the first thing we’re going to do is build a #site-wrapper div that wraps everything just after the body tag. The site wrapper will be used to hide overflowed elements without messing up the scroll. We’ll go over that shortly.

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title></title>

</head>

<body>

<div id="site-wrapper">

</div>

</body>

</html>

The next thing we’re going to need to do add is our#site-canvas wrapper inside of our #site-wrapper. The canvas part is what slides left and right and holds all the content of the page.

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title></title>

</head>

<body>

<div id="site-wrapper">

<div id="site-canvas">

</div>

</div>

</body>

</html>

Another piece that we need to add is the #site-menu. From the blue print above, it might be a little deceiving. We’re actually going to put the menu inside of #site-canvas despite that it appears outside of the element. The reason we do this is because we want the menu to slide when we move the canvas (again, which basically holds everything). Later, we’ll position *absolute* the menu to appear outside of the canvas container.

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title></title>

</head>

<body>

<div id="site-wrapper">

<div id="site-canvas">

<div id="site-menu">

</div>

</div>

</div>

</body>

</html>

The last necessary element that we need to add is an atag to toggle the event of opening and closing the nav.

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title></title>

</head>

<body>

<div id="site-wrapper">

<div id="site-wrapper">

<div id="site-menu">

<a href="#" class="toggle-nav">Toggle Nav</a>

</div>

</div>

</div>

</body>

</html>

The demo I’m doing also uses Bootstrap 3 (for speeding things up) and Font Awesome (for aesthetics), but neither are required for the tutorial. Below is it all together with some extra markup for the demo. The highlighted lines are the important stuff.

<!doctype html>

<html>

<head>

<!-- Meta -->

<meta charset="utf-8">

<title>Off Canvas Menus Demo</title>

<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<!-- Favicon -->

<link rel="shortcut icon" sizes="16x16 24x24 32x32 48x48 64x64" href="<https://scotch.io/favicon.ico>">

<!-- Styles -->

<link rel="stylesheet" href="//netdna.bootstrapcdn.com/bootstrap/3.1.0/css/bootstrap.min.css">

<link rel="stylesheet" href="//netdna.bootstrapcdn.com/font-awesome/4.0.3/css/font-awesome.css">

<link rel="stylesheet" href="/path/to/your/custom/style.css">

<!--[if lt IE 9]><script src="//html5shiv.googlecode.com/svn/trunk/html5.js"></script><![endif]-->

</head>

<body>

<div id="site-wrapper">

<div id="site-canvas">

<div id="site-menu">

<h2>My Menu</h2>

<p class="lead">Put any HTML you want here.

Style it however you want.

<ul>

<li>Free to scroll up and down</li>

<li>But not left and write</li>

</ul>

</div>

<a href="#" class="toggle-nav btn btn-lg btn-success" id="big-sexy"><i class="fa fa-bars"></i> Toggle Nav</a>

<h1 class="text-center">Off Canvas Menu Tutorial</h1>

<p class="lead text-center">Basic demo showing how to easily implement an Off Canvas Menu by <a href="[https://scotch.io](https://scotch.io/)">scotch.io</a>.

<!-- #site-canvas -->

</div>

<!-- #site-wrapper> -->

</div>

<!-- Scripts -->

<script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js"></script>

<script src="/path/to/your/custom/power.js"></script>

</body>

</html>

<

div class=”alert alert-warning”>  
**Heads Up!** If you’re not operating this from a web server (aka your computer locally without a virtual environment), you’ll need to change any CDN references from “**//**” to “**https://**” or “**http://**“.

[**The CSS**](https://scotch.io/tutorials/off-canvas-menus-with-css3-transitions-and-transforms#the-css)

**The Wrapper**

The wrapper, in this case #site-wrapper, is used to hide elements that overflow outside of it. The reason we need to hide overflowed elements on a wrapper instead of the body is so that we don’t mess up scrolling. Try putting overflow: hidden; on a body tag and see what happens to scrolling.

By having the wrapper, you can actually specify how large you want the Off Canvas Menu block to be. This example is going to be full width, but as shown from the small demos above, you can essentially make it any size you want.

The only thing weird that we’re going to do for this demo is put a fixed height on this element. We’re only doing this to mimic that we actually have content that fills our page.

#site-wrapper {

position: relative;

overflow: hidden;

width: 100%;

height: 5000px; /\* Temp: Simulates a tall page. \*/

}

**The “Canvas”**

The #site-canvas is the actual element moving left and right. We’ll set a *transition* on it for doing a smooth animation, and we’ll also be just setting the *transform*property in preparation for movement. The technique we are using uses CSS3 Transforms to move the canvas instead of JavaScript. This technique makes it smoother and sexier.

#site-canvas {

width: 100%;

height: 100%;

position: relative;

-webkit-transform: translateX(0);

transform: translateX(0);

-webkit-transition: .3s ease all;

transition: .3s ease all;

padding: 5% 0; /\* Temp: Just spacing. \*/

}

**The Menu**

Remember how the menu is inside of the canvas element, but positioned outside of it? Well, lets go ahead and add the CSS to make that happen:

#site-menu {

width: 300px;

height: 100%;

position: absolute;

top: 0;

left: -300px;

background: #428bca;

padding: 15px;

}

**The Animation**

The idea behind the animation is simple. When you want the menu to slide out, we’ll add a class to the wrapper, then just add properties doing a smooth transition to slide the canvas over.

#site-wrapper.show-nav #site-canvas {

-webkit-transform: translateX(300px);

transform: translateX(300px);

}

In order to add that class to the wrapper, we’ll have to use JavaScript though.

[**The JavaScript**](https://scotch.io/tutorials/off-canvas-menus-with-css3-transitions-and-transforms#the-javascript)

We need to be able to toggle the .show-nav class on the wrapper whenever we want to reveal the navigation per a certain event (which is going to be a click). We’ll use the class .toggle-nav to detect click events. This way you can put the class on as many different a tags as you want (e.g.: You can place an a tag that says “Close” inside of the navigation to toggle the nav off so long that it has a class of .toggle-nav). We’ll be using [jQuery](http://jquery.com/)to speed things up.

/\*====================================

= ON DOM READY =

====================================\*/

$(function() {

$('.toggle-nav').click(function() {

// Calling a function in case you want to expand upon this.

toggleNav();

});

});

/\*========================================

= CUSTOM FUNCTIONS =

========================================\*/

function toggleNav() {

if ($('#site-wrapper').hasClass('show-nav')) {

// Do things on Nav Close

$('#site-wrapper').removeClass('show-nav');

} else {

// Do things on Nav Open

$('#site-wrapper').addClass('show-nav');

}

//$('#site-wrapper').toggleClass('show-nav');

}

You can expand or rewrite this however you’d like, but the main thing is to make sure that the wrapper class is toggling. Depending on the site, I’ll sometimes put the.show-nav class on the body so I have full control to tweak any element on the page via CSS while the navigation is open if I need to.

[**The Result**](https://scotch.io/tutorials/off-canvas-menus-with-css3-transitions-and-transforms#the-result)

[**Tricks and Tips to Supercharge**](https://scotch.io/tutorials/off-canvas-menus-with-css3-transitions-and-transforms#tricks-and-tips-to-supercharge)

**Leverage the computer’s GPU**

Since we are using *translates* to animate the opening and closing of the Off Canvas Menu, there’s some somewhat unknown CSS3 tricks to help improve performance. If you use a 3D translate and the computer has it, you’ll force the users computer to use hardware acceleration to make the animation smoother and faster. The difference might be very subtle, but for some users and machines it will make the world of difference. Here’s the code:

/\* Original Method \*/

#site-canvas {

-webkit-transform: translateX(0);

transform: translateX(0);

}

.show-nav #site-canvas {

-webkit-transform: translateX(300px);

transform: translateX(300px);

}

/\* Better Performing Method \*/

#site-canvas {

-webkit-transform: translate3d(0);

transform: translate3d(0);

}

.show-nav #site-canvas {

-webkit-transform: translate3d(300px, 0, 0);

transform: translate3d(300px, 0, 0);

}

Another trick you could do to force hardware acceleration is to use <i>translateZ</i>. Here's an example to best explain:

/\* Original Method \*/

#site-canvas {

-webkit-transform: translateX(0);

transform: translateX(0);

}

.show-nav #site-canvas {

-webkit-transform: translateX(300px);

transform: translateX(300px);

}

/\* Better Performing Method \*/

#site-canvas {

-webkit-transform: translateZ(0) translateX(0);

transform: translateZ(0) translateX(0);

}

.show-nav #site-canvas {

-webkit-transform: translateZ(0) translateX(300px);

transform: translateZ(0) translateX(300px);

}

**Use Backface-Visibility**

A lot of transforms and 3D transforms are still experimental in browsers. If you’re getting weird flickering or DOM painting issues, try adding it to the#site-canvas.

#site-canvas {

-webkit-backface-visibility: hidden;

backface-visibility: hidden;

}

**Make the escape key close the nav**

Depending on how big your Off Canvas Menu is, it might make sense to add this feature in for your users. It’s good usability and makes a lot sense. You can add the following JavaScrip to close the nav when users hit the esc key.

$(document).keyup(function(e) {

if (e.keyCode == 27) {

if ($('#site-wrapper').hasClass('show-nav')) {

// Assuming you used the function I made from the demo

toggleNav();

}

}

});

**Enable Fast Click for Mobile**

An Off Canvas Menu works great on mobile, but most mobile devices have a lag when you click a link. This is frustrating for the user and a laggy experience overall. The easiest way is to use this [FastClick](https://github.com/ftlabs/fastclick) JavaScript library. If you don’t want to use another JavaScript library, here’s a tutorial by Google on [mobile fast click](https://developers.google.com/mobile/articles/fast_buttons).

**Use Different Types of Transitions**

The CSS3 transition property is what makes the animation smooth. If you didn’t include it, the #site-canvas would almost instantaneously be pushed over with no known animation of it happening. The transition property has a lot of cool features beyond just speeding it up and slowing it down. Here’s some examples to try:

/\*

\* Ease

\* Default and very smooth transition.

\*/

#site-canvas {

-webkit-transition: 300ms ease all;

transition: 300ms ease all;

}

/\*

\* Linear

\* Similar to if you would animate with jQuery (without ease)

\*/

#site-canvas {

-webkit-transition: 300ms linear all;

transition: 300ms linear all;

}

/\*

\* Ease-in

\* Accelerates

\*/

#site-canvas {

-webkit-transition: 300ms ease-in all;

transition: 300ms ease-in all;

}

/\*

\* Ease-out

\* Decelerates

\*/

#site-canvas {

-webkit-transition: 300ms ease-out all;

transition: 300ms ease-out all;

}

/\*

\* Ease-in-out

\* Slow, fast, slow

\*/

#site-canvas {

-webkit-transition: 300ms ease-out all;

transition: 300ms ease-out all;

}

/\*==================================================

= CUBIC - Custom Transitions =

==================================================\*/

/\*

\* Cubic

\* Bounce

\*/

#site-canvas {

-webkit-transition: all 300ms cubic-bezier(0.32,1.25,0.375,1.15);

transition: all 300ms cubic-bezier(0.32,1.25,0.375,1.15);

}

/\*

\* Cubic

\* Snappy Acceleration

\*/

#site-canvas {

-webkit-transition: -webkit-transform .33s cubic-bezier(.694, .0482, .335, 1);

transition: -webkit-transform .33s cubic-bezier(.694, .0482, .335, 1);

}

/\*

\* Cubic

\* Out of Orbit

\*/

#site-canvas {

-webkit-transition: -webkit-transform .33s cubic-bezier(1,0,.61,.15);

transition: -webkit-transform .33s cubic-bezier(1,0,.61,.15);

}

/\*

\* Cubic

\* Fat kid down slide

\*/

#site-canvas {

-webkit-transition: -webkit-transform .33s cubic-bezier(.2,1,.47,0);

transition: -webkit-transform .33s cubic-bezier(.2,1,.47,0);

}

**Darken Overlay or close on clicking the canvas**

You can easily append this code to put a dark, 30% opacity or so overlay over the #site-canvas on open. You can even add a click event to the overlay to toggle the navigation off. This tutorial won’t go over how to do that, but it’s good practice to give the users as much usability as possible.

[**Browser Support Help**](https://scotch.io/tutorials/off-canvas-menus-with-css3-transitions-and-transforms#browser-support-help)

It’s important to support multiple browsers. Thankfully the use case for Off Canvas Menus are generally smart phones and tablets which come with fairly modern browsers; however, you should still try and support many devices and systems. The example above only works in Chrome, but there’s many ways to extend support to other browsers. Here’s a list of properties you’ll need to cross-support:

* [3D Transforms](http://caniuse.com/transforms3d)
* [2D Transforms](http://caniuse.com/transforms2d)
* [Transitions](http://caniuse.com/css-transitions)

**3D Translate with a 2D Fallback**

Using *translate3d* is well worth it in terms of performance and user experience. However, not all browsers support it. Here’s how you can gracefully degrade the Transform for maximum browser support. Pretty straight-forward:

#site-canvas {

-webkit-transform: translateX(0);

transform: translateX(0);

-webkit-transform: translateX(0) translateZ(0);

transform: translateX(0) translateZ(0);

-webkit-transform: translate3d(0, 0, 0);

transform: translate3d(0, 0, 0);

}

.show-nav #site-canvas {

-webkit-transform: translateX(300px);

transform: translateX(300px);

-webkit-transform: translateX(300px) tranlateZ(0);

transform: translateX(300px) tranlateZ(0);

-webkit-transform: translate3d(300px, 0, 0);

transform: translate3d(300px, 0, 0);

}

**Browser Prefixes**

I left these out of the example to keep things easy on the eye, but here’s what an example would look like with maximum browser support.

#site-canvas {

-moz-transform: translateX(0);

-ms-transform: translateX(0);

-o-transform: translateX(0);

-webkit-transform: translateX(0);

transform: translateX(0);

-moz-transform: translateZ(0) translateX(0);

-ms-transform: translateZ(0) translateX(0);

-o-transform: translateZ(0) translateX(0);

-webkit-transform: translateZ(0) translateX(0);

transform: translateZ(0) translateX(0);

-moz-transform: translate3d(0, 0, 0);

-ms-transform: translate3d(0, 0, 0);

-o-transform: translate3d(0, 0, 0);

-webkit-transform: translate3d(0, 0, 0);

transform: translate3d(0, 0, 0);

-moz-transition: 300ms ease all;

-ms-transition: 300ms ease all;

-o-transition: 300ms ease all;

-webkit-transition: 300ms ease all;

transition: 300ms ease all;

-moz-backface-visibility: hidden;

-ms-backface-visibility: hidden;

-o-backface-visibility: hidden;

-webkit-backface-visibility: hidden;

backface-visibility: hidden;

}

.show-nav #site-canvas {

-moz-transform: translateX(300px);

-ms-transform: translateX(300px);

-o-transform: translateX(300px);

-webkit-transform: translateX(300px);

transform: translateX(300px);

-moz-transform: translateZ(0) translateX(300px);

-ms-transform: translateZ(0) translateX(300px);

-o-transform: translateZ(0) translateX(300px);

-webkit-transform: translateZ(0) translateX(300px);

transform: translateZ(0) translateX(300px);

-moz-transform: translate3d(300px, 0, 0);

-ms-transform: translate3d(300px, 0, 0);

-o-transform: translate3d(300px, 0, 0);

-webkit-transform: translate3d(300px, 0, 0);

transform: translate3d(300px, 0, 0);

}

**JavaScript Fallback**

As a last resort you, you could always implement a JavaScript fallback. For example:

function toggleNav() {

if ($('#site-wrapper').hasClass('show-nav')) {

// Logic here if browser doesn't support/have CSS3 Transforms

$('#site-wrapper').css('margin-right', '0px');

$('#site-wrapper').removeClass('show-nav');

} else {

// Logic here if browser doesn't support/have CSS3 Transforms

$('#site-wrapper').css('margin-right', '-300px');

$('#site-wrapper').addClass('show-nav');

}

//$('#site-wrapper').toggleClass('show-nav');

}

[**Conclusion**](https://scotch.io/tutorials/off-canvas-menus-with-css3-transitions-and-transforms#conclusion)

Even though this tutorial only covered one style of Off Canvas Menu with multiple types of *transitions*, you can play around with it to make menus however you’d like. For example, resize the scotch.io site and look at how our Off Canvas Menu works. Another personal favorite to check out is [SquareSpace](http://squarespace.com/) and how they did their menu. The options really are limitless.

I hope you enjoyed the tutorial and demos. Thanks for checking it out! I’m always looking for feedback, improvements, and suggestions.