Introduction

Get started with fancybox, probably the world's most popular lightbox script.

Dependencies

jQuery 3+ is preferred, but fancybox works with jQuery 1.9.1+ and jQuery 2+

Compatibility

fancybox includes support for touch gestures and even supports pinch gestures for zooming. It is perfectly suited for both mobile and desktop browsers.

fancybox has been tested in following browsers/devices:

- Chrome
- Firefox
- IE10/11
- Edge
- iOS Safari
- Android 7.0 Tablet

Setup

You can install fancybox by linking `.css` and `.js` files to your html file. Make sure you also load the jQuery library. Below is a basic HTML template to use as an example:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>My page</title>

    <!-- CSS -->
    <link rel="stylesheet" type="text/css" href="jquery.fancybox.min.css">
</head>
<body>

    <!-- Your HTML content goes here -->

    <!-- JS -->
    <script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
    <script src="jquery.fancybox.min.js"></script>
</body>
</html>
```

Download fancybox

Download the latest version of fancybox on GitHub.
Or just link directly to fancybox files on cdnjs - https://cdnjs.com/libraries/fancybox.

Package Managers

fancybox is also available on npm and Bower.

```
# NPM
npm install @fancyapps/fancybox --save

# Bower
bower install fancybox --save
```

Important

- Make sure you add the jQuery library before the fancybox JS file
- If you already have jQuery on your page, you shouldn't include it second time
- Do not include both fancybox.js and fancybox.min.js files
- Some functionality (ajax, iframes, etc) will not work when you're opening local file directly on your browser, the code must be running on a web server

How to Use

Initialize with data attributes

The most basic way to use fancybox is by adding the `data-fancybox` attribute to your element. This will automatically bind click event that will start fancybox. Use `href` or `data-src` attribute to specify source of your content. Example:

```
<a href="image.jpg" data-fancybox data-caption="Caption for single image">
    <img src="thumbnail.jpg" alt="" />
</a>
```

View demo on CodePen

If you have a group of items, you can use the same attribute `data-fancybox` value for each of them to create a gallery. Each group should have a unique value. Example:

```
<a href="image_1.jpg" data-fancybox="gallery" data-caption="Caption #1">
    <img src="thumbnail_1.jpg" alt="" />
</a>

<a href="image_2.jpg" data-fancybox="gallery" data-caption="Caption #2">
    <img src="thumbnail_2.jpg" alt="" />
</a>
```

View demo on CodePen

If you choose this method, default settings will be applied. See options section for examples how to customize by changing defaults, using `data-options` attribute or by initializing with JavaScript.

Info Sometimes you have multiple links pointing to the same source and that creates duplicates in the gallery. To avoid that, simply use `data-fancybox-trigger` attribute with the same value used for `data-fancybox` attribute for your other links. Optionally, use `data-fancybox-index` attribute to specify index of starting element:

```
<a data-fancybox-trigger="gallery" href="javascript:;">
    <img src="thumbnail_1.jpg" alt="" />
</a>
```

View demo on CodePen

Initialize with JavaScript

Select your elements with a jQuery selector (you can use any valid selector) and call the `fancybox` method:

```
$('[data-fancybox="gallery"]').fancybox({
    // Options will go here
});
```

View demo on CodePen

Info Sometimes you might need to bind fancybox to dynamically added elements. Use `selector` option to attach click event listener for elements that exist now or in the future. All selected items will be automatically grouped in the gallery. Example:

```
$().fancybox({
    selector : '.imglist a:visible'
});
```

View demo on CodePen

Use with Javascript

You can also open and close fancybox programmatically. Here are a couple of examples, visit API section for more information and demos.

Display simple message:

```
$.fancybox.open('<div class="message"><h2>Hello!</h2><p>You are awesome!</p></div>');
```

View demo on CodePen

Display iframed page:

```
$.fancybox.open({
    src  : 'link-to-your-page.html',
    type : 'iframe',
    opts : {
        afterShow : function( instance, current ) {
            console.info( 'done!' );
        }
    }
});
```

View demo on CodePen

Important

fancybox attempts to automatically detect the type of content based on the given url. If it cannot be detected, the type can also be set manually using `data-type` attribute (or `type` option). Example:

```
<a href="images.php?id=123" data-type="image" data-caption="Caption">
    Show image
</a>
```

Media types

fancybox is designed to display images, video, iframes and any HTML content. For your convenience, there is a built in support for inline content and ajax.

Images

The standard way of using fancybox is with a number of thumbnail images that link to larger images:

```
<a href="image.jpg" data-fancybox="images" data-caption="My caption">
    <img src="thumbnail.jpg" alt="" />
</a>
```

View demo on CodePen

By default, fancybox fully preloads an image before displaying it. You can choose to display the image right away. It will render and show the full size image while the data is being received. To do so, some attributes are necessary:

- `data-width` - the real width of the image
- `data-height` - the real height of the image

```
<a href="image.jpg" data-fancybox="images" data-width="2048" data-height="1365">
    <img src="thumbnail.jpg" />
</a>
```

View demo on CodePen

You can also use these `width` and `height` properties to control size of the image. This can be used to make images look sharper on retina displays. Example:

```
$('[data-fancybox="images"]').fancybox({
    afterLoad : function(instance, current) {
        var pixelRatio = window.devicePixelRatio || 1;

        if ( pixelRatio > 1.5 ) {
            current.width  = current.width  / pixelRatio;
            current.height = current.height / pixelRatio;
        }
    }
});
```

View demo on CodePen

fancybox supports "srcset" so it can display different images based on viewport width. You can use this to improve download times for mobile users and over time save bandwidth. Example:

```
<a href="medium.jpg" data-fancybox="images" data-srcset="large.jpg 1600w, medium.jpg
1200w, small.jpg 640w">
    <img src="thumbnail.jpg" />
</a>
```

View demo on CodePen

It is also possible to protect images from downloading by right-click. While this does not protect from truly determined users, it should discourage the vast majority from ripping off your files. Optionally, put the watermark over image.

```
$('[data-fancybox]').fancybox({
    protect: true
});
```

View demo on CodePen

Video

YouTube and Vimeo videos can be used with fancybox by just providing the page URL. Link to MP4 video directly or use trigger element to display hidden `<video>` element.

Use `data-width` and `data-height` attributes to customize video dimensions and `data-ratio` for the aspect ratio.

```
<a data-fancybox href="https://www.youtube.com/watch?v=_sI_Ps7JSEk">
    YouTube video
</a>

<a data-fancybox href="https://vimeo.com/191947042">
    Vimeo video
</a>

<a data-fancybox data-width="640" data-height="360" href="video.mp4">
    Direct link to MP4 video
</a>

<a data-fancybox href="#myVideo">
    HTML5 video element
</a>

<video width="640" height="320" controls id="myVideo" style="display:none;">
    <source src="https://www.html5rocks.com/en/tutorials/video/basics/Chrome_ImF.mp4"
type="video/mp4">
    <source src="https://www.html5rocks.com/en/tutorials/video/basics/Chrome_ImF.webm"
type="video/webm">
    <source src="https://www.html5rocks.com/en/tutorials/video/basics/Chrome_ImF.ogv"
type="video/ogg">
    Your browser doesn't support HTML5 video tag.
</video>
```

View demo on CodePen

Controlling YouTube & Vimeo video via URL parameters:

```
<a data-fancybox href="https://www.youtube.com/watch?
v=_sI_Ps7JSEk&amp;autoplay=1&amp;rel=0&amp;controls=0&amp;showinfo=0">
    YouTube video - hide controls and info
</a>

<a data-fancybox href="https://vimeo.com/191947042?color=f00">
    Vimeo video - custom color
</a>
```

View demo on CodePen

Via JavaScript:

```
$('[data-fancybox]').fancybox({
    youtube : {
        controls : 0,
        showinfo : 0
    },
    vimeo : {
        color : 'f00'
    }
});
```

View demo on CodePen

Iframe

If you need to display content from another page, add `data-fancybox` and `data-type="iframe"` attributes to your link. This would create `<iframe>` element that allows to embed an entire web document inside the modal.

```
<a data-fancybox data-type="iframe" data-src="http://codepen.io/fancyapps/full/jyEGGG/"
href="javascript:;">
    Webpage
</a>

<a data-fancybox data-type="iframe" data-
src="https://mozilla.github.io/pdf.js/web/viewer.html" href="javascript:;">
    Sample PDF file
</a>
```

View demo on CodePen

If you have not disabled iframe preloading (using `preload` option), the script will atempt to calculate content dimensions and will adjust width/height of `<iframe>` to fit with content in it. Keep in mind, that due to same origin policy, there are some limitations.

This example will disable iframe preloading and will display small close button next to iframe instead of the toolbar:

```
$('[data-fancybox]').fancybox({
    toolbar  : false,
    smallBtn : true,
    iframe : {
        preload : false
    }
})
```

View demo on CodePen

Iframe dimensions can be controlled by CSS:

```
.fancybox-slide--iframe .fancybox-content {
    width  : 800px;
    height : 600px;
    max-width  : 80%;
    max-height : 80%;
    margin: 0;
}
```

These CSS rules can be overridden by JS, if needed:

```
$("[data-fancybox]").fancybox({
    iframe : {
        css : {
            width : '600px'
        }
    }
});
```

How to access and control fancybox in parent window from inside an iframe:

```
// Close current fancybox instance
parent.jQuery.fancybox.getInstance().close();

// Adjust iframe height according to the contents
parent.jQuery.fancybox.getInstance().update();
```

Inline

fancybox can be used to display any HTML element on the page. First, create a hidden element with unique ID:

```
<div style="display: none;" id="hidden-content">
    <h2>Hello</h2>
    <p>You are awesome.</p>
</div>
```

Then simply create a link having `data-src` attribute that matches ID of the element you want to open (preceded by a hash mark (#); in this example - `#hidden-content`):

```
<a data-fancybox data-src="#hidden-content" href="javascript:;">
    Trigger the fancybox
</a>
```

View demo on CodePen

The script will append small close button (if you have not disabled by `smallBtn:false`) and will not apply any styles except for centering. Therefore you can easily set custom dimensions using CSS.

Info If necessary, you can make your element (and similarly any other html content) scrollable by adding additional wrapping element and some CSS - view demo on CodePen.

Ajax

To load your content via AJAX, you need to add a `data-type="ajax"` attribute to your link:

```
<a data-fancybox data-type="ajax" data-src="my_page.com/path/to/ajax/"
href="javascript:;">
    AJAX content
</a>
```

View demo on CodePen

Additionally it is possible to define a selector with the `data-filter` attribute to show only a part of the response. The selector can be any string, that is a valid jQuery selector:

```
<a data-fancybox data-type="ajax" data-src="my_page.com/path/to/ajax/" data-
filter="#two" href="javascript:;">
    AJAX content
</a>
```

View demo on CodePen

Options

Quick reference for all default options as defined in the source:

```
var defaults = {
  // Close existing modals
  // Set this to false if you do not need to stack multiple instances
  closeExisting: false,

  // Enable infinite gallery navigation
  loop: false,

  // Horizontal space between slides
  gutter: 50,

  // Enable keyboard navigation
  keyboard: true,

  // Should allow caption to overlap the content
  preventCaptionOverlap: true,

  // Should display navigation arrows at the screen edges
  arrows: true,

  // Should display counter at the top left corner
  infobar: true,

  // Should display close button (using `btnTpl.smallBtn` template) over the content
  // Can be true, false, "auto"
  // If "auto" - will be automatically enabled for "html", "inline" or "ajax" items
  smallBtn: "auto",

  // Should display toolbar (buttons at the top)
  // Can be true, false, "auto"
  // If "auto" - will be automatically hidden if "smallBtn" is enabled
  toolbar: "auto",

  // What buttons should appear in the top right corner.
  // Buttons will be created using templates from `btnTpl` option
  // and they will be placed into toolbar (class="fancybox-toolbar"` element)
  buttons: [
    "zoom",
    //"share",
    "slideShow",
    //"fullScreen",
    //"download",
    "thumbs",
    "close"
  ],

  // Detect "idle" time in seconds
  idleTime: 3,

  // Disable right-click and use simple image protection for images
```

Set instance options by passing a valid object to `fancybox()` method:

```
$('[data-fancybox="gallery"]').fancybox({
    thumbs : {
        autoStart : true
    }
});
```

Plugin options / defaults are exposed in `$.fancybox.defaults` namespace so you can easily adjust them globally:

```
$.fancybox.defaults.animationEffect = "fade";
```

Custom options for each element individually can be set by adding a `data-options` attribute to the element. This attribute should contain the properly formatted JSON object (remember, strings should be wrapped in double quotes).

It is also possible to quickly set any option using *parameterized* name of the selected option, for example, `animationEffect` would be `data-animation-effect`:

```
<a data-fancybox data-options='{"caption" : "My caption", "src" :
"https://codepen.io/about/", "type" : "iframe"}' href="javascript:;" class="btn btn-
primary">
    Example #1
</a>

<a data-fancybox data-animation-effect="false"
href="https://source.unsplash.com/0JYgd2QuMfw/1500x1000" class="btn btn-primary">
    Example #2
</a>
```

View demo on CodePen

API

The fancybox API offers a couple of methods to control fancybox. This gives you the ability to extend the plugin and to integrate it with other web application components.

Core methods

Core methods are methods which affect/handle instances:

```
// Start new fancybox instance
$.fancybox.open( items, opts, index );

// Get refrence to currently active fancybox instance
$.fancybox.getInstance();

// Close currently active fancybox instance (pass `true` to close all instances)
$.fancybox.close();

// Close all instances and unbind all events
$.fancybox.destroy();
```

Starting fancybox

When creating group objects manually, each item should follow this pattern:

```
{
    src  : '' // Source of the content
    type : '' // Content type: image|inline|ajax|iframe|html (optional)
    opts : {} // Object containing item options (optional)
}
```

Example of opening image gallery programmatically:

```
$.fancybox.open([
    {
        src  : '1_b.jpg',
        opts : {
            caption : 'First caption',
            thumb   : '1_s.jpg'
        }
    },
    {
        src  : '2_b.jpg',
        opts : {
            caption : 'Second caption',
            thumb   : '2_s.jpg'
        }
    }
], {
    loop : false
});
```

View demo on CodePen

It is also possible to pass only one object. Example of opening inline content:

```
$.fancybox.open({
    src  : '#hidden-content',
    type : 'inline',
    opts : {
        afterShow : function( instance, current ) {
            console.info( 'done!' );
        }
    }
});
```

View demo on CodePen

If you wish to quickly display some html content (for example, a message), then you can use a simpler syntax. Do not forget to use a wrapping element around your content.

```
$.fancybox.open('<div class="message"><h2>Hello!</h2><p>You are awesome!</p></div>');
```

View demo on CodePen

Group items can be collection of jQuery objects, too. This can be used, for example, to display group of inline elements:

```
$('#test').on('click', function() {
  $.fancybox.open( $('.inline-gallery'), {
    touch: false
  });
});
```

View demo on CodePen

Instance methods

In order to use these methods, you need an instance of the plugin's object. There are 3 common ways to get the reference.

1) Using API method to get currently active instance:

```
var instance = $.fancybox.getInstance();
```

2) While starting fancybox programmatically:

```
var instance = $.fancybox.open(
    // Your content and options
);
```

3) From within the callback - first argument is always a reference to current instance:

```
$('[data-fancybox="gallery"]').fancybox({
    afterShow : function( instance, current ) {
        console.info( instance );
    }
});
```

Once you have a reference to fancybox instance the following methods are available:

```
// Go to next gallery item
instance.next( duration );

// Go to previous gallery item
instance.previous( duration );

// Switch to selected gallery item
instance.jumpTo( index, duration );

// Check if current image dimensions are smaller than actual
instance.isScaledDown();

// Scale image to the actual size of the image
instance.scaleToActual( x, y, duration );

// Check if image dimensions exceed parent element
instance.canPan();

// Scale image to fit inside parent element
instance.scaleToFit( duration );

// Update position and content of all slides
instance.update();

// Update slide position and scale content to fit
instance.updateSlide( slide );

// Update infobar values, navigation button states and reveal caption
instance.updateControls( force );

// Load custom content into the slide
instance.setContent( slide, content );

// Show loading icon inside the slide
instance.showLoading( slide );

// Remove loading icon from the slide
instance.hideLoading( slide );

// Try to find and focus on the first focusable element
instance.focus();

// Activates current instance, brings it to the front
instance.activate();

// Close instance
instance.close();
```

You can also do something like this:

```
$.fancybox.getInstance().jumpTo(1);
```

or simply:

```
$.fancybox.getInstance('jumpTo', 1);
```

Events

fancybox fires several events:

```
beforeLoad   : Before the content of a slide is being loaded
afterLoad    : When the content of a slide is done loading

beforeShow   : Before open animation starts
afterShow    : When content is done loading and animating

beforeClose  : Before the instance attempts to close. Return false to cancel the close.
afterClose   : After instance has been closed

onInit       : When instance has been initialized
onActivate   : When instance is brought to front
onDeactivate : When other instance has been activated
```

Event callbacks can be set as function properties of the options object passed to fancybox initialization function:

```
<script type="text/javascript">
    $("[data-fancybox]").fancybox({
        afterShow: function( instance, slide ) {

            // Tip: Each event passes useful information within the event object:

            // Object containing references to interface elements
            // (background, buttons, caption, etc)
            // console.info( instance.$refs );

            // Current slide options
            // console.info( slide.opts );

            // Clicked element
            // console.info( slide.opts.$orig );

            // Reference to DOM element of the slide
            // console.info( slide.$slide );

        }
    });
</script>
```

Each callback receives two parameters - current fancybox instance and current gallery object, if exists.

It is also possible to attach event handler for all instances. To prevent interfering with other scripts, these events have been namespaced to `.fb`. These handlers receive 3 parameters - event, current fancybox instance and current gallery object.

Here is an example of binding to the `afterShow` event:

```
$(document).on('afterShow.fb', function( e, instance, slide ) {
    // Your code goes here
});
```

If you wish to prevent closing of the modal (for example, after form submit), you can use `beforeClose` callback. Simply return `false`:

```
beforeClose : function( instance, current, e ) {
    if ( $('#my-field').val() == '' ) {
        return false;
    }
}
```

Modules

fancybox code is split into several files (modules) that extend core functionality. You can build your own fancybox version by excluding unnecessary modules, if needed. Each one has their own `js` and/or `css` files.

Some modules can be customized and controlled programmatically. List of all possible options:

```
fullScreen: {
  autoStart: false
},

touch : {
  vertical : true,  // Allow to drag content vertically
  momentum : true   // Continuous movement when panning
},

// Hash value when initializing manually,
// set `false` to disable hash change
hash : null,

// Customize or add new media types
// Example:
/*
media : {
  youtube : {
    params : {
      autoplay : 0
    }
  }
}
*/
media : {},

slideShow : {
  autoStart : false,
  speed     : 4000
},

thumbs : {
  autoStart   : false,                  // Display thumbnails on opening
  hideOnClose : true,                   // Hide thumbnail grid when closing animation
starts
  parentEl    : '.fancybox-container',  // Container is injected into this element
  axis        : 'y'                     // Vertical (y) or horizontal (x) scrolling
},

share: {
  url: function(instance, item) {
    return (
      (!instance.currentHash && !(item.type === "inline" || item.type === "html") ?
item.origSrc || item.src : false) || window.location
    );
  },
  tpl:
    '<div class="fancybox-share">' +
    "<h1>{{SHARE}}</h1>" +
    "<p>" +
```

Couple of examples

Show thumbnails on start:

```
$('[data-fancybox="images"]').fancybox({
    thumbs : {
        autoStart : true
    }
});
```

View demo on CodePen

Customize share url if displaying hidden video element:

```
$('[data-fancybox="test-share-url"]').fancybox({
    buttons : ['share', 'close'],
    hash : false,
    share : {
        url : function( instance, item ) {
            if (item.type === 'inline' && item.contentType === 'video') {
                return item.$content.find('source:first').attr('src');
            }

            return item.src;
        }
    }
});
```

View demo on CodePen

If you would inspect fancybox instance object, you would find that same keys ar captialized - these are references for each module object. Also, you would notice that fancybox uses common naming convention to prefix jQuery objects with `$`.

This is how you, for example, can access thumbnail grid element:

```
$.fancybox.getInstance().Thumbs.$grid
```

This example shows how to call method that toggles thumbnails:

```
$.fancybox.getInstance().Thumbs.toggle();
```

List of available methods:

```
Thumbs.focus()
Thumbs.update();
Thumbs.hide();
Thumbs.show();
Thumbs.toggle();

FullScreen.request( elem );
FullScreen.exit();
FullScreen.toggle( elem );
FullScreen.isFullscreen();
FullScreen.enabled();

SlideShow.start();
SlideShow.stop();
SlideShow.toggle();
```

If you wish to disable hash module, use this snippet (after including JS file):

```
$.fancybox.defaults.hash = false;
```

FAQ

#1 Opening/closing causes fixed element to jump

Simply add `compensate-for-scrollbar` CSS class to your fixed positioned elements. Example of using Bootstrap navbar component:

```
<nav class="navbar navbar-inverse navbar-fixed-top compensate-for-scrollbar">
    <div class="container">
        ..
    </div>
</nav>
```

View demo on CodePen

The script measures width of the scrollbar and creates `compensate-for-scrollbar` CSS class that uses this value for `margin-right` property. Therefore, if your element has `width:100%`, you should positon it using `left` and `right` properties instead. Example:

```
.navbar {
    position: fixed;
    top: 0;
    left: 0;
    right: 0;
}
```

#2 How to customize caption

You can use `caption` option that accepts a function and is called for each group element. Example of appending image download link:

```
$( '[data-fancybox="images"]' ).fancybox({
    caption : function( instance, item ) {
        var caption = $(this).data('caption') || '';

        if ( item.type === 'image' ) {
            caption = (caption.length ? caption + '<br />' : '') + '<a href="' +
item.src + '">Download image</a>' ;
        }

        return caption;
    }
});
```

View demo on CodePen

Add current image index and image count (the total number of images in the gallery) right in the caption:

```
$( '[data-fancybox="images"]' ).fancybox({
    infobar : false,
    caption : function( instance, item ) {
        var caption = $(this).data('caption') || '';

        return ( caption.length ? caption + '<br />' : '' ) + 'Image <span data-
fancybox-index></span> of <span data-fancybox-count></span>';
    }
});
```

View demo on CodePen

Inside `caption` method, `this` refers to the clicked element. Example of using different source for caption:

```
$( '[data-fancybox]' ).fancybox({
    caption : function( instance, item ) {
        return $(this).find('figcaption').html();
    }
});
```

View demo on CodePen

#3 How to create custom button in the toolbar

Example of creating reusable button:

```
// Create template for the button
$.fancybox.defaults.btnTpl.fb = '<button data-fancybox-fb class="fancybox-button
fancybox-button--fb" title="Facebook">' +
    '<svg viewBox="0 0 24 24">' +
        '<path d="M22.676 0H1.324C.594 0 0 .593 0 1.324v21.352C0 23.408.593 24 1.324
24h11.494v-9.294h-3.13v-3.62h3.13V8.41c0-3.1 1.894-4.785 4.66-4.785 1.324 0 2.463.097
2.795.14v3.24h-1.92c-1.5 0-1.793.722-1.793 1.772v2.31h3.584l-.465 3.63h-
3.12V24h6.115c.733 0 1.325-.592 1.325-1.324V1.324C24 .594 23.408 0 22.676 0"/>' +
    '</svg>' +
'</button>';

// Make button clickable using event delegation
$('body').on('click', '[data-fancybox-fb]', function() {
    window.open("https://www.facebook.com/sharer/sharer.php?
u="+encodeURIComponent(window.location.href)+"&t="+encodeURIComponent(document.title),
'','left=0,top=0,width=600,height=300,menubar=no,toolbar=no,resizable=yes,scrollbars=yes');

});

// Customize buttons
$( '[data-fancybox="images"]' ).fancybox({
    buttons : [
        'fb',
        'close'
    ]
});
```

View demo on CodePen

#4 How to reposition thumbnail grid

There is currenty no JS option to change thumbnail grid position. But fancybox is designed so that you can use CSS to change position or dimension for each block (e.g., content area, caption or thumbnail grid). This gives you freedom to completely change the look and feel of the modal window, if needed.
View demo on CodePen

#5 How to disable touch gestures/swiping

When you want to make your content selectable or clickable, you have two options:

- disable touch gestures completely by setting `touch:false`
- add `data-selectable="true"` attribute to your html element

View demo on CodePen

#6 Slider/carousel add's cloned duplicate items

If you are combining fancybox with slider/carousel script and that script clones items to enable infinite navigation, then duplicated items will appear in the gallery. To avoid that - 1) initialise fancybox on all items except cloned; 2) add custom click event on cloned items and trigger click event on corresponding "real" item. Here is an example using Slick slider:

```
// Init fancybox
// =============
var selector = '.slick-slide:not(.slick-cloned)';

// Skip cloned elements
$().fancybox({
  selector : selector,
  backFocus : false
});

// Attach custom click event on cloned elements,
// trigger click event on corresponding link
$(document).on('click', '.slick-cloned', function(e) {
  $(selector)
    .eq( ( $(e.currentTarget).attr("data-slick-index") || 0) % $(selector).length )
    .trigger("click.fb-start", {
      $trigger: $(this)
    });

  return false;
});

// Init Slick
// ==========
$(".main-slider").slick({
  slidesToShow   : 3,
  infinite : true,
  dots     : false,
  arrows   : false
});
```

View demo on CodePen

Back to Top