# Tutorial - Rich Text Inline Edit

This page will walk you through an example of adding the Fluid Rich Text Inline Edit component to an HTML file. For more general information about the Inline Edit API, see Rich Text Inline Edit API.

This tutorial assumes that:

- you are already familiar with HTML, Javascript and CSS
- you are familiar with what the Inline Edit is and does
- now you just want to know how to add it to your file.

## Tutorial: How to Use the Inline Edit

### Scenario

You've created a database to keep track of your vast collection of CDs, and you're working on a web interface for it. When viewing the details of a CD, you would like to very easily add a personal review. Prone to rambling, you know you will want to add some basic styling so the text is easier to read. This tutorial will show you how to use the Fluid Rich Text Inline Edit for this.

There are three basic steps to adding the Inline Edit to your application:

- Setup: Download and install the Fluid Infusion library
- Step 1: Prepare your Markup
- Step 2: Write the script
- Step 3: Add the required libraries to your HTML

The rest of this tutorial will explain each of these steps in detail.

| **Status** |
|---|
| This component is in Sneak Peek status |

**On This Page**

**See Also**

- Inline Edit
- Simple Text Inline Edit API

**Still need help?**

Join the infusion-users mailing list and ask your questions there.

---

## Setup: Download and install the Fluid Infusion library

1. Download a copy of the Fluid Infusion component library from:
    - http://fluidproject.org/products/fluid-infusion/download-infusion/
      You only really need the "Minified deployment package," but if you want to actually look at the code, you should download the "Source package."
2. Unpack the zip file you just downloaded, and place the resulting folder somewhere convenient for your development purposes.
   The folder will have the release number in its name (e.g. infusion-1.4/). The rest of this tutorial will use infusion-1.4 in its examples, but if you downloaded a different version, you'll have to adjust.

---

## Step 1: Prepare your markup

Let's assume that you're working with some HTML that displays the information about a CD in your collection - something simple like this:

```
<h1>Album Details</h1>

<h2>Artist</h2>
Portishead

<h2>Album</h2>
Third

<h2>Year</h2>
2008

<h2>Label</h2>
Universal Music

<h2>Review</h2>
<p>After a hiatus, <strong>Portishead</strong> is back with their first studio
album in 6 years. <em>Third</em> brings back the familiar and
the new, and none of this is best exemplified than in the track
<em>Machine Gun</em>.</p>

<p>It seems that regardless of how <strong>Portishead</strong> sounds now, the one
thing that has stayed constant is their refusal to be ordinary.</p>
```

### The Changes

In order to apply the Rich Text editor to your markup, you need to tell it three things:

1. which element is the block of text you want to make editable by identifying a container,
2. how you want the rich inline edit component to look by using a another container, and
3. group the above two containers inside a parent container.

If we want to make the review text element editable, then we could

1. wrap entire review text inside a `<div>` element and give it the default class `flc-inlineEdit-text`.
2. create a new `<div>` element with the default class name `flc-inlineEdit-editContainer` for the editor.
3. place the above two `<div>` containers inside a `<div>` and give it a unique ID.

We can also define how big we want the rich edit field to be, and add *Save* and *Cancel* buttons.

This might look like the HTML sample to the right.

```
<h1>Album Details</h1>

<h2>Artist</h2>
Portishead

<h2>Album</h2>
Third

<h2>Year</h2>
2008

<h2>Label</h2>
Universal Music

<h2>Review</h2>
<div id="cd-review">
  <div class="flc-inlineEdit-text">
    <p>After a hiatus, <strong>Portishead</strong> is back with their first studio
    album in 6 years. <em>Third</em> brings back the familiar and
    the new, and none of this is best exemplified than in the track
    <em>Machine Gun</em>.</p>

    <p>It seems that regardless of how <strong>Portishead</strong> sounds now, the one
    thing that has stayed constant is their refusal to be ordinary.</p>
  </div>

  <div class="flc-inlineEdit-editContainer">
    <textarea rows="10" cols="80" ></textarea>
    <button class="save">Save</button> <button class="cancel">Cancel</button>
  </div>
</div>
```

That's all - these are the only changes you need to make to your HTML.

---

## Step 2: Write the script

There are some scripts you will need to add to your HTML file before the Rich Text Inline Edit will function properly.

### Define Button Behaviour

The `<button>` elements by themselves will not do anything special unless we specify some behaviour. This can be done by adding the following script:

```
function makeButtons(editor) {
  $(".save", editor.container).click(function(){
    editor.finish();
    return false;
  });

  $(".cancel", editor.container).click(function(){
    editor.cancel();
    return false;
  });
}
```

This will make the `Save` and `Cancel` buttons perform what we'd expect.

### Initialize the Rich Text Inline Editor

Now you need to initialize the Rich Text Inline Editor.

If using **TinyMCE**, this can be accomplished by the following code:

```
var richEditor = fluid.inlineEdit.tinyMCE("#cd-review", {tinyMCE: {width: 1024}});
makeButtons(richEditor);
```

Note: We specify an optional width for the TinyMCE editor so that it fits the width of the container more closely.

If using **CKEditor**, this can be accomplished by the following code:

```
var richEditor = fluid.inlineEdit.CKEditor("#cd-review");
makeButtons(richEditor);
```

This function (regardless of the editor being used) will look inside the element with the "cd-review" ID (in this case, your `<div>` element) for anything with the `flc-inlineEdit-text` and `flc-inlineEdit-editContainer` class, and convert it into an Rich Text Inline Edit field.

Note: The FCKEditor integration is deprecated for v1.2. Please use the CKEditor integration instead.

### Putting it All Together

Combining all of the above scripts together, it will look like this:

```
function makeButtons(editor) {
  $(".save", editor.container).click(function(){
    editor.finish();
    return false;
  });

  $(".cancel", editor.container).click(function(){
    editor.cancel();
    return false;
  });
}

var richEditor = fluid.inlineEdit.tinyMCE("#cd-review", {tinyMCE: {width: 1024}});
makeButtons(richEditor);
```

By putting these functions inside the `jQuery(document).ready()` call, you ensure that all of your markup is ready before you try to initialize the Rich Text Inline Edit components. This script can also be placed in a `<script>` block at the end of your document.

---

## Step 3: Add the Fluid library to your HTML

You'll need to add the Fluid library to your HTML file. In the header of the file, link to the Javascript files with `<script>` tags:

**CKEditor Integration:**

```
<script type="text/javascript" src="http://ckeditor-fluid.appspot.com/ckeditor.js"></script>
<script type="text/javascript" src="InfusionAll.js"></script>
```

**TinyMCE Integration:**

```
<script type="text/javascript" src="http://tinymce-fluid.appspot.com/tiny_mce.js"></script>
<script type="text/javascript" src="InfusionAll.js"></script>
```

Alternatively, the individual file requirements if using CKEditor are:

```
<script type="text/javascript" src="http://ckeditor-fluid.appspot.com/ckeditor.js"></script>
<script type="text/javascript" src="lib/jquery/core/js/jquery.js"></script>
<script type="text/javascript" src="lib/jquery/ui/js/ui.core.js"></script>
<script type="text/javascript" src="lib/jquery/plugins/tooltip/js/jquery.tooltip.js"></script>
<script type="text/javascript" src="framework/core/js/jquery.keyboard-a11y.js"></script>
<script type="text/javascript" src="framework/core/js/Fluid.js"></script>
<script type="text/javascript" src="components/inlineEdit/js/InlineEdit.js"></script>
<script type="text/javascript" src="components/inlineEdit/js/InlineEditIntegrations.js"></script>
```

Otherwise, the individual file requirements if using TinyMCE are:

```
<script type="text/javascript" src="http://tinymce-fluid.appspot.com/tiny_mce.js"></script>
<script type="text/javascript" src="lib/jquery/core/js/jquery.js"></script>
<script type="text/javascript" src="lib/jquery/ui/js/ui.core.js"></script>
<script type="text/javascript" src="lib/jquery/plugins/tooltip/js/jquery.tooltip.js"></script>
<script type="text/javascript" src="framework/core/js/jquery.keyboard-a11y.js"></script>
<script type="text/javascript" src="framework/core/js/Fluid.js"></script>
<script type="text/javascript" src="components/inlineEdit/js/jquery.tinymce.js"></script>
<script type="text/javascript" src="components/inlineEdit/js/InlineEdit.js"></script>
<script type="text/javascript" src="components/inlineEdit/js/InlineEditIntegrations.js"></script>
```