

Setting the Body in the Applet Tutorial

Getting Started

Required Skills

The following skills are required prior to working with this tutorial:

- Basic client-side JavaScript

Required Tutorials Completed

The following tutorials are required to be undertaken before attempting this tutorial:

- [Instantiation Tutorial](#)

Tutorial

Step 1. Create an Instance of EditLive! in a Webpage

As shown in the [Instantiation Tutorial](#), create an instance of EditLive! in a webpage.

```
<html>
  <body>
    <script src="../../redistributables/editlivejava/editlivejava.js" language="JavaScript"></script>
    <script>
      var editlivejava = new EditLiveJava("ELJApplet", "700", "400");
      editlivejava.setConfigurationFile("../../redistributables/editlivejava/sample_eljconfig.
xml");

      editlivejava.show();
    </script>
  </body>
</html>
```

Save this webpage as *setBody.html*.

Step 2. Setting the <BODY> to Appear When the Editor Loads

The [body](#) load-time property is used to set the contents of the <BODY> element for the HTML Document stored in EditLive!.

For the purpose of this tutorial, we will load the following HTML into the editor:

<p>Original <i>HTML</i> loaded into EditLive!</p>



When specifying any HTML to insert into EditLive! for Java, this HTML must be URL Encoded. Javascript provides numerous URL Encoding methods (such as *escape* and *encodeURIComponent*). If using a server-side language to create the webpage where EditLive! is instantiated, Ephox advises using the URL encoding method of this language to encode the HTML to be used with either the load-time properties or run-time functions of EditLive!.

```
<html>
  <body>
    <script src="../../redistributables/editlivejava/editlivejava.js" language="JavaScript"></script>
    <script>
      var editlivejava = new EditLiveJava("ELJApplet", "700", "400");
      editlivejava.setConfigurationFile("../../redistributables/editlivejava/sample_eljconfig.
xml");

      editlivejava.setBody(encodeURIComponent("<p>Original <i>HTML</i> loaded into EditLive!<
/p>"));

      editlivejava.show();
    </script>
  </body>
</html>
```

Step 3. Create a HTML Textarea

Add a HTML textarea to the webpage. The purpose of this textarea will be to allow users to write HTML content, which they can then send to EditLive! at any time.

```
<html>
  <body>
    <p><textarea id="bodyContents" cols="80" rows="5"></textarea></p>
    <script src="../../redistributables/editlivejava/editlivejava.js" language="JavaScript"></script>
    <script>
      var editlivejava = new EditLiveJava("ELJApplet", "700", "400");
      editlivejava.setConfigurationFile("../../redistributables/editlivejava/sample_eljconfig.
xml");
      editlivejava.setBody(encodeURIComponent("<p>Original <i>HTML</i> loaded into EditLive!<
/p>"));

      editlivejava.show();
    </script>
  </body>
</html>
```

Step 4. Add a Button to Copy the Textarea Contents into EditLive!

In order to copy the contents of the textarea into EditLive!, a button should be created to call a Javascript method. This Javascript method will be written in the next step to perform the copying action.

```
<html>
  <body>
    <p><textarea id="bodyContents" cols="80" rows="5"></textarea>
      <br/><input type="button" value="Set <BODY> Contents" onclick="buttonPress()"></p>
    <script src="../../redistributables/editlivejava/editlivejava.js" language="JavaScript"></script>
    <script>
      var editlivejava = new EditLiveJava("ELJApplet", "700", "400");
      editlivejava.setConfigurationFile("../../redistributables/editlivejava/sample_eljconfig.
xml");
      editlivejava.setBody(encodeURIComponent("<p>Original <i>HTML</i> loaded into EditLive!<
/p>"));

      editlivejava.show();
    </script>
  </body>
</html>
```

Step 5. Create a Javascript Function to Set the <BODY> of EditLive! with the Textarea Contents

The [setBody Method](#) can be used to the <BODY> of EditLive!. You can directly reference the textarea's content by using the Javascript DOM methods.

As seen in Step 2, a URL encoding function is required when passing HTML to EditLive!.

```

<html>
  <body>
    <form name="exampleForm">
      <p><textarea id="bodyContents" cols="80" rows="5"></textarea>
        <br/><input type="button" value="Set <BODY> Contents" onclick="buttonPress()"><
      /p>
      <script src="../../redistributables/editlivejava/editlivejava.js" language="JavaScript"><
    /script>
      <script>
        var editlivejava = new EditLiveJava("ELJApplet", "700", "400");
        editlivejava.setConfigurationFile("../../redistributables/editlivejava
      /sample_eljconfig.xml");
        editlivejava.setBody(encodeURIComponent("<p>Original <i>HTML</i> loaded into
      EditLive!</p>"));

        editlivejava.show();

        function buttonPress() {
          editlivejava.setBody(encodeURIComponent(document.exampleForm.
      bodyContents.value));
        }
      </script>
    </form>
  </body>
</html>

```

See Also

- [<sourceEditor>](#) Configuration Element
- [setBody Method](#)
- [getBody Method](#)