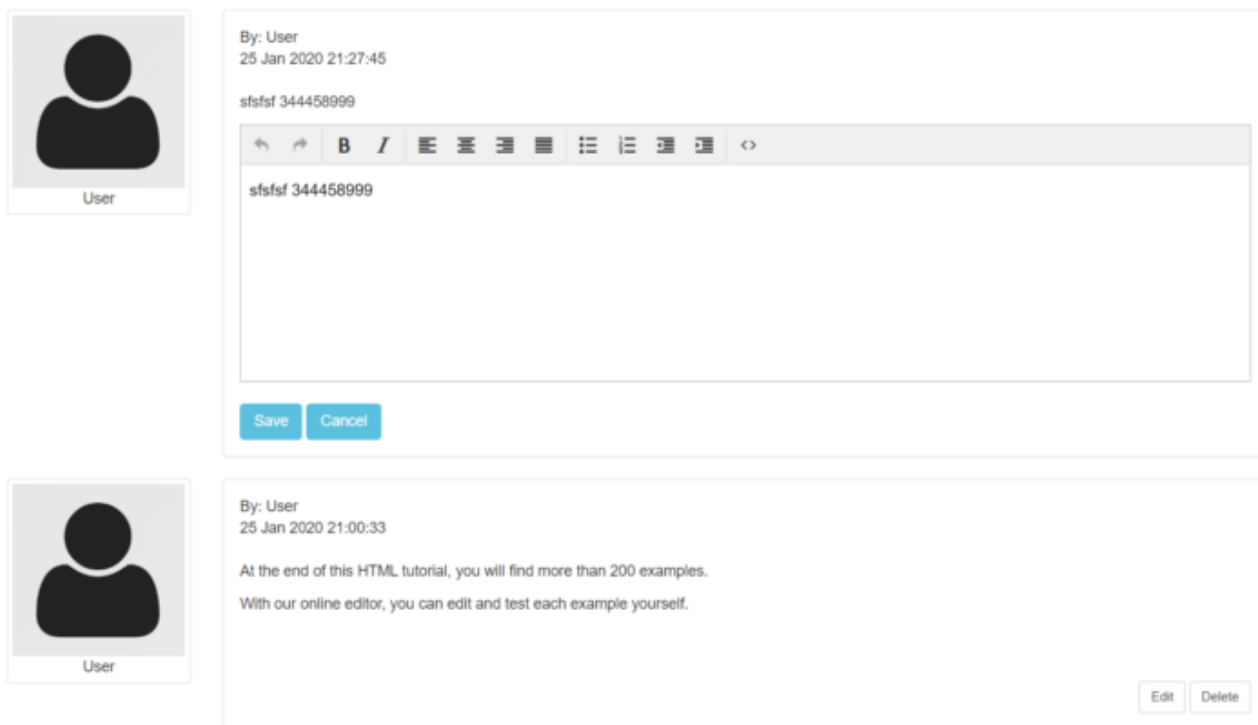


Загрузка динамического редактора TinyMCE с использованием Ajax, PHP и MySQL.

В предыдущем уроке мы объяснили, как реализовать загрузку изображений в TinyMCE с помощью Ajax и PHP. В этом уроке вы узнаете, как загружать динамические изображения в редакторе TinyMCE с помощью Ajax, PHP и MySQL.

TinyMCE — популярный WYSIWYG-редактор, используемый для ввода данных с целью их сохранения в базе данных. Он применяется в веб-приложениях для добавления и редактирования HTML-контента. Новый контент добавляется с помощью фиксированного редактора, но редактор TinyMCE необходимо динамически загружать и удалять для редактирования контента из разных записей.

Если вы ищете решения для динамической загрузки и удаления редактора TinyMCE для сохранения содержимого, то вы попали по адресу. В этом руководстве вы узнаете, как динамически загружать и удалять редактор TinyMCE для редактирования и сохранения содержимого из разных записей с помощью Ajax, PHP и MySQL.



The image displays two user profile cards. Each card features a placeholder image for a user profile and a text area for content. The top card shows a text area with the placeholder text "sfsfsf 344458999" and a toolbar with various editing options. The bottom card shows a text area with the text "At the end of this HTML tutorial, you will find more than 200 examples. With our online editor, you can edit and test each example yourself." and buttons for "Edit" and "Delete".

В этом пошаговом руководстве мы рассмотрим, как динамически загружать и удалять редактор TinyMCE, а также сохранять контент в таблицу базы данных MySQL.

Итак, давайте реализуем динамическую загрузку и удаление редактора TinyMCE и сохранение контента в таблицу базы данных MySQL с помощью Ajax, PHP и MySQL. Основные файлы:

- **index.php** : PHP-файл для загрузки редактора TinyMCE и записи сообщений.
- **tinymce_editor.js** : JavaScript-файл для инициализации редактора TinyMCE и обеспечения функциональности сохранения изменений.
- **action.php** : PHP-файл для обработки AJAX-запросов на редактирование и сохранение записей.

- **Message.php** : Класс, содержащий методы для получения, сохранения и редактирования записей.

Таблица базы данных

Как показано в этом руководстве, мы будем отображать записи из базы данных и редактировать/сохранять их, загружая динамический редактор TinyMCE. Поэтому сначала мы создадим таблицу в базе данных MySQL, **message** используя приведенное ниже выражение для создания таблицы, чтобы реализовать эту функциональность.

MySQL

```
CREATE TABLE `message` (  
  `id` int(11) NOT NULL,  
  `message` text NOT NULL,  
  `user` varchar(255) NOT NULL,  
  `created` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;  
  
ALTER TABLE `message`  
  ADD PRIMARY KEY (`id`);  
  
ALTER TABLE `message`  
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
```

Шаг 1: Подключите редактор TinyMCE и jQuery.

Мы загрузим библиотеку редактора TinyMCE, сохраним её в папке tinymce и включим в index.php файл. Также мы подключим jQuery и пользовательский JavaScript-tinymce_editor.js файл.

tinymce_editor.js

```
<script  
src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">  
</script>  
<script src="tinymce/tinymce.min.js"></script>  
<script src="js/tinymce_editor.js"></script>
```

Шаг 2: Загрузите редактор TinyMCE, чтобы сохранить

новое сообщение.

Мы создадим форму с текстовым полем для загрузки редактора TinyMCE, чтобы добавить новое сообщение.

[index.php](#)

```
<form id="posts" name="posts" method="post">
  <textarea name="message" id="message"></textarea><br>
  <input type="hidden" name="action" id="action" value="save" />
  <button type="submit" id="save" name="save" class="btn btn-info
saveButton">Сохранить</button>
</form>
```

Шаг 3: Загрузите редактор TinyMCE.

В `tinymce_editor.js` файле мы создадим функцию `startEditor()` для инициализации редактора TinyMCE.

[tinymce_editor.js](#)

```
function startEditor(id) {
  tinymce.init({
    selector: "textarea#" + id,
    plugins: [
      "code ",
      "paste"
    ],
    toolbar: "undo redo | bold italic | alignleft aligncenter
alignright alignjustify | bullist numlist outdent indent | link code ",
    menubar: false,
    statusbar: false,
    content_style: ".mce-content-body {font-size:15px;font-
family:Arial,sans-serif;}",
    height: 200
  });
}
```

Мы вызовем функцию `startEditor()` и укажем идентификатор текстового поля, чтобы загрузить редактор.

[.js](#)

```
$( document ).ready(function() {
  startEditor('message');
```

```
});
```

Шаг 4: Список записей с сообщением

В `index.php` файле мы выведем список сообщений с кнопками редактирования для изменения текста сообщения.

`index.php`

```
<div id="postList">
<?php
$result = $message->getPost();
while ($post = $result->fetch_assoc()) {
    $date = date_create($post['created']);
    ?>
    <article class="row" id="postRow_<?php echo $post['id']; ?>">
        <div class="col-md-2 col-sm-2 hidden-xs">
            <figure class="thumbnail">
                
                <figcaption class="text-center"><?php echo
ucwords($post['user']); ?></figcaption>
            </figure>
        </div>
        <div class="col-md-10 col-sm-10">
            <div class="panel panel-default arrow left">
                <div class="panel-body">
                    <header class="text-left">
                        <div class="comment-user"><i class="fa fa-user"></i>
Автор: <?php echo $post['user']; ?></div>
                        <time class="comment-date" datetime="16-12-2014
01:05"><i class="fa fa-clock-o"></i> <?php echo date_format($date, 'd
MYH:i:s'); ?></time>
                    </header>
                    <div class="comment-post" id="post_message_<?php echo
$post['id']; ?>">

                        <?php echo $post['message']; ?>

                    </div>

                    <textarea name="message" id="<?php echo $post['id']; ?>"
style="visibility:hidden;"></textarea><br>

                    <div class="text-right" id="button_section_<?php echo
$post['id']; ?>">
                        <a class="btn btn-default btn-sm" id="edit_<?php echo
$post['id']; ?>"><i class="fa fa-reply"></i> Редактировать</a>
```

```

        <a class="btn btn-default btn-sm"><i class="fa fa-
reply"></i> Удалить</a>
    </div>

    <div id="editSection_<?php echo $post['id']; ?>"
class="hidden">
        <button type="submit" id="save_<?php echo $post['id'];
?>" name="save" class="btn btn-info saveButton">Сохранить</button>
        <button type="submit" id="cancel_<?php echo
$post['id']; ?>" name="cancel" class="btn btn-info
saveButton">Отменить</button>
    </div>
</div>
</div>
</article>
<?php } ?>
</div>

```

Шаг 5: Загрузите динамический редактор TinyMCE для редактирования сообщения.

В `tinymce_editor.js` файле мы загрузим TinyMCE, вызвав функцию `startEditor(messageId)` при нажатии кнопки редактирования сообщения. Мы загрузим редактор TinyMCE вместе с сообщением.

[tinymce_editor.js](#)

```

$('#postList').on('click', '[id^=edit_]', function(event){
    var messageId = $(this).attr('id');
    messageId = messageId.replace(/edit_/g, '');
    messageId = parseInt(messageId);
    tinymce.execCommand("mceRemoveEditor", true, messageId);
    startEditor(messageId);
    tinymce.get(messageId).setContent($("#post_message_"+messageId).html());
    ;
    $("#editSection_"+messageId).removeClass('hidden');
    $("#button_section_"+messageId).addClass('hidden');
});

```

Шаг 6: Редактировать сообщение о сохранении

В `tinymce_editor.js` файле мы реализуем функциональность сохранения сообщения из

динамического редактора с помощью Ajax-запроса `action.php` с действием `update` редактирования/сохранения сообщения. Мы также обеспечим отображение обновленного сообщения с подробной информацией.

tinymce_editor.js

```
$('#postLsit').on('click', '[id^=save_]', function(event){
    var messageId = $(this).attr('id');
    messageId = messageId.replace(/save_/g, '');
    messageId = parseInt(messageId);
    var postMessage = tinymce.get(messageId).getContent();
    tinymce.execCommand("mceRemoveEditor", true, messageId);
    var action = 'update';
    $.ajax({
        url: 'action.php',
        method: "POST",
        data: {id: messageId, message: postMessage, action: action},
        dataType: "json",
        success: function(data){
            var html = $("#postHtml").html();
            html = html.replace(/USERNAME/g, data.user);
            html = html.replace(/POSTDATE/g, data.post_date);
            html = html.replace(/POSTMESSAGE/g, data.message);
            html = html.replace(/POSTID/g, data.id);
            $("#postRow_"+messageId).html(html);
            $("#editSection_"+messageId).addClass('hidden');
            $("#button_section_"+messageId).removeClass('hidden');
        }
    });
});
```

В `action.php` файле мы проверим наличие действия `update` и вызовем `update()` метод из класса `Message.php` для обновления записи.

action.php

```
if(!empty($_POST['message']) && $_POST['message'] && $_POST['action']
== 'update') {
    $message->message = $_POST['message'];
    $message->id = $_POST['id'];
    $message->update();
}
```

На занятии `Message.php` мы реализуем `update()` метод для обновления сообщения в таблице базы данных и возврата обновленных сведений о записи.

Message.php

```
public function update(){

    if($this->id && $this->message) {

        $stmt = $this->conn->prepare("
            UPDATE ".$this->messageTable." SET message = ?
            WHERE id = ?");

        $stmt->bind_param("si", $this->message, $this->id);

        if($stmt->execute()){
            $sqlQuery = "
                SELECT id, message, user, DATE_FORMAT(created,'%d %M %Y
%H:%i:%s') AS post_date
                FROM ".$this->messageTable." WHERE id =
'".$this->id."'";
            $stmt2 = $this->conn->prepare($sqlQuery);
            $stmt2->execute();
            $result = $stmt2->get_result();
            $record = $result->fetch_assoc();
            echo json_encode($record);
        }
    }
}
```

Шаг 7: Удалите динамический редактор TinyMCE.

В `tinymce_editor.js` файле мы реализуем функциональность для удаления загруженного редактора TinyMCE при нажатии кнопки «Отмена».

[tinymce_editor.js](#)

```
$('#postLsit').on('click', '[id^=cancel_]', function(event){
    var messageId = $(this).attr('id');
    messageId = messageId.replace(/cancel_/g, '');
    tinymce.execCommand("mceRemoveEditor", true, messageId);
    $("#editSection_"+messageId).addClass('hidden');
    $("#button_section_"+messageId).removeClass('hidden');
});
```

Дополнения и Файлы

- [Ссылка на оригинальную статью](#)
- [dynamic-tinymce-editor_1.zip](#)

From:
<https://wwoss.direct.quickconnect.to/> - **worldwide open-source software**

Permanent link:
https://wwoss.direct.quickconnect.to/doku.php?id=software:development:web:docs:web:wysiwyg:tinymce-editor_ajax_php_mysql&rev=1767955519

Last update: **2026/01/09 13:45**

