


This repository

Search or type a command

ExploreGistBlogHelp

Macro-Bin



nswbmw / N-blog

Watch92Star387Fork245

第5章 增加编辑与删除功能

Edit PagePage HistoryClone URL

现在，我们来给博客添加编辑文章与删除文章的功能。

我们设定：当一个用户在线时，只允许他在自己发表的文章页进行编辑或删除，编辑时，只能编辑文章内容，不能编辑文章标题。

打开 `style.css`，添加如下样式：

```
.edit{margin:3px;padding:2px 5px;border-radius:3px;background-color:#f3f3f3;color:#333;font-size:13px;}
.edit:hover{text-decoration:none;background-color:#f00;color:#fff;-webkit-transition:color .2s linear;}
```

打开 `article.ejs`，将代码修改成如下：

```
<%- include header %>
<p>
  <span><a class="edit" href="/edit/<%= post.name %>/<%= post.time.day %>/<%= post.title %>">编辑</a></span>
  <span><a class="edit" href="/remove/<%= post.name %>/<%= post.time.day %>/<%= post.title %>">删除</a></span>
</p>
<p class="info">
  作者: <a href="/u/<%= post.name %>"><%= post.name %></a> |
  日期: <%= post.time.minute %>
</p>
<p><%= post.post %></p>
<%- include footer %>
```

至此，我们只是在文章页面添加了编辑和删除文章的链接。接下来，我们注册这两个链接的响应。

打开 `post.js`，在最后添加如下代码：

```
//返回原始发表的内容（markdown 格式）
Post.edit = function(name, day, title, callback) {
  //打开数据库
  mongodb.open(function (err, db) {
    if (err) {
      return callback(err);
    }
    //读取 posts 集合
    db.collection('posts', function (err, collection) {
      if (err) {
        mongodb.close();
        return callback(err);
      }
      //根据用户名、发表日期及文章名进行查询
      collection.findOne({
        "name": name,
        "time.day": day,
        "title": title
      }, function (err, doc) {
        mongodb.close();
        if (err) {
          return callback(err);
        }
        callback(null, doc);//返回查询的一篇文章（markdown 格式）
      });
    });
  });
};
```

打开 `index.js`，在 `app.get('/u/:name/:day/:title')` 后添加如下代码：

```
app.get('/edit/:name/:day/:title', checkLogin);
app.get('/edit/:name/:day/:title', function (req, res) {
  var currentUser = req.session.user;
```

```
Post.edit(currentUser.name, req.params.day, req.params.title, function (err, post) {
  if (err) {
    req.flash('error', err);
    return res.redirect('back');
  }
  res.render('edit', {
    title: '编辑',
    post: post,
    user: req.session.user,
    success: req.flash('success').toString(),
    error: req.flash('error').toString()
  });
});
```

在 `views` 下新建 `edit.ejs`，添加如下代码：

```
<%- include header %>
<form method="post">
  标题: <br />
  <input type="text" name="title" value="<%= post.title %>" disabled="disabled" /><br />
  正文: <br />
  <textarea name="post" rows="20" cols="100"><%= post.post %></textarea><br />
  <input type="submit" value="保存修改" />
</form>
<%- include footer %>
```

运行我们的博客看看吧。至此，在文章页面，当我们点击 `编辑` 链接后就会跳转到该文章对应的编辑页面了。接下来我们实现将修改后的文章提交到数据库。

打开 `post.js`，在最后添加如下代码：

```
//更新一篇文章及其相关信息
Post.update = function(name, day, title, post, callback) {
  //打开数据库
  mongodb.open(function (err, db) {
    if (err) {
      return callback(err);
    }
    //读取 posts 集合
    db.collection('posts', function (err, collection) {
      if (err) {
        mongodb.close();
        return callback(err);
      }
      //更新文章内容
      collection.update({
        "name": name,
        "time.day": day,
        "title": title
      }, {
        $set: {post: post}
      }, function (err, result) {
        mongodb.close();
        if (err) {
          return callback(err);
        }
        callback(null);
      });
    });
  });
};
```

打开 `index.js`，在 `app.get('/edit/:name/:day/:title')` 后添加如下代码：

```
app.post('/edit/:name/:day/:title', checkLogin);
app.post('/edit/:name/:day/:title', function (req, res) {
  var currentUser = req.session.user;
  Post.update(currentUser.name, req.params.day, req.params.title, req.body.post, function (err) {
    var url = '/u/' + req.params.name + '/' + req.params.day + '/' + req.params.title;
    if (err) {
      req.flash('error', err);
    }
  });
});
```

```
    return res.redirect(url);//出错! 返回文章页
  }
  req.flash('success', '修改成功!');
  res.redirect(url);//成功! 返回文章页
});
});
```

现在,我们就可以编辑并保存文章了。赶紧试试吧!

接下来,我们实现删除文章的功能。打开 `post.js`, 在最后添加如下代码:

```
//删除一篇文章
Post.remove = function(name, day, title, callback) {
  //打开数据库
  mongodb.open(function (err, db) {
    if (err) {
      return callback(err);
    }
    //读取 posts 集合
    db.collection('posts', function (err, collection) {
      if (err) {
        mongodb.close();
        return callback(err);
      }
      //根据用户名、日期和标题查找并删除一篇文章
      collection.remove({
        "name": name,
        "time.day": day,
        "title": title
      }, function (err, result) {
        mongodb.close();
        if (err) {
          return callback(err);
        }
        callback(null);
      });
    });
  });
};
```

打开 `index.js`, 在 `app.post('/edit/:name/:day/:title')` 后添加如下代码:

```
app.get('/remove/:name/:day/:title', checkLogin);
app.get('/remove/:name/:day/:title', function (req, res) {
  var currentUser = req.session.user;
  Post.remove(currentUser.name, req.params.day, req.params.title, function (err) {
    if (err) {
      req.flash('error', err);
      return res.redirect('back');
    }
    req.flash('success', '删除成功!');
    res.redirect('/');
  });
});
```

至此我们完成了大部分的工作,接下来我们实现页面权限的控制。假如你现在注册了两个帐号 **A** 和 **B**, 那么当 **B** 访问 **A** 的用户页面时,也会出现编辑和删除的选项,虽然点击后并不能编辑和删除 **A** 的文章。那怎么才能实现一个账号只能编辑和删除自己发表的文章呢?很简单,添加一个判断即可。打开 `article.js`, 将:

```
<span><a class="edit" href="/edit/<%= post.name %>/<%= post.time.day %>/<%= post.title %>">编辑</a></span>
<span><a class="edit" href="/remove/<%= post.name %>/<%= post.time.day %>/<%= post.title %>">删除</a></span>
```

修改为:

```
<% if (user && (user.name == post.name)) { %>
  <span><a class="edit" href="/edit/<%= post.name %>/<%= post.time.day %>/<%= post.title %>">编辑</a></span>
  <span><a class="edit" href="/remove/<%= post.name %>/<%= post.time.day %>/<%= post.title %>">删除</a></span>
<% } %>
```

以上代码通过检测 `session` 中的用户名是否存在,若存在该用户和当前进入的用户页面的用户名相同,相同则显示编辑和删除按钮,不同则不显示。

现在，我们完成了给博客添加编辑文章与删除文章的功能。

Last edited by nswbmw, a month ago

