导航菜单管理

导航菜单的数据会以 JSON 的方式存在 `options` 表中 `key` 为 `nav_menus` 的 `value` 中。

1. 加载数据到表格展示

> 一般系统的 `options` 表的结构就是键值结构，也就是必然会有两个列，一个是 `key`，另一个是 `value`，这种结构比较灵活

在页面加载过后，根据配置选项的键 `nav_menus` 获取对应的数据（JSON）：

```
$(function () {
  /**
   * 显示消息
   * @param  {String} msg 消息文本
   */
  function notify (msg) {
    $('.alert').text(msg).fadeIn()
    // 3000 ms 后隐藏
    setTimeout(function () {
      $('.alert').fadeOut()
    }, 3000)
  }

  /**
   * 加载导航菜单数据
   */
  function loadData () {
    $.get('/admin/options.php', { key: 'nav_menus' }, function (res) {
      if (!res.success) {
        // 失败，提示
        return notify(res.message)
      }

      var menus = []

      try {
        // 尝试以 JSON 方式解析响应内容
        menus = JSON.parse(res.data)
      } catch (e) {
        notify('获取数据失败')
      }

      // 使用 jsrender 渲染数据到表格
      $('tbody').html($('#menu_tmpl').render(menus))
    })
  }

  // 首次加载数据
  loadData()
})
```

源代码: step-76

--

2. 新增导航菜单

思路：在点击保存按钮时，先获取全部导航菜单的数据，然后将界面上填写的数据 push 进去，然后再序列化为一个 JSON 字符串，通过 AJAX 发送到服务端保存。

> 名词解释：
>
> 1. 将一个对象转换为一个 JSON 字符串的过程叫做**序列化**；
> 2. 同理将一个 JSON 字符串转换为一个对象的过程叫做**反序列化**；

2.1. 获取当前导航菜单数据

作为当前的情况，我们可以有两种方式获取当前导航菜单数据：

1. 将之前的 menus 定义成全局成员，让其在按钮点击时可以被访问。
2. 点击时再次发送 AJAX 请求获取**最新**的数据。

> 提问：哪一种方式跟合适？

发送异步请求：

之前我们已经定义了一个加载数据的 `loadData` 函数，但是在这里不能共用，因为在这个函数中拿到数据过后就渲染到界面上了，而我们这里是需要这个数据做后续逻辑。

如果需要公用，则需要改造这个函数，让其返回数据，而不是使用数据。

> 函数的粒度问题 函数的粒度指的是在同一个函数中业务的数量。
>
> 1. 粒度越细，公用性越好
> 2. 粒度越粗，调用越方便，性能大多数越好。

返回数据的方式：

如果是一个普通情况下的函数数据返回，直接使用 `return` 即可，但是此处我们的数据是需要 AJAX 过后才能拿到的，不能使用简单的 `return` 返回，即异步编程最常见的问题，必须使用回调（委托）解决。

重新封装 `loadData()`：

```
/**
 * 加载导航菜单数据
 * @param {Function} callback 获取到数据后续的逻辑
 */
function loadData (callback) {
  $.get('/admin/options.php', { key: 'nav_menus' }, function (res) {
    if (!res.success) {
      // 失败，提示
      return callback(new Error(res.message))
    }

    var menus = []

    try {
      // 尝试以 JSON 方式解析响应内容
      menus = JSON.parse(res.data)
    } catch (e) {
      callback(new Error('获取数据失败'))
    }

    callback(null, menus)
  })
}
```

首次加载数据时：

```
// 首次加载数据
loadData(function (err, data) {
  if (err) return notify(err.message)
  // 使用 jsrender 渲染数据到表格
  $('tbody').html($('#menu_tmpl').render(data))
})
```

> 源代码: step-77

```
/**
 * 新增逻辑
 */
$('.btn-save').on('click', function () {
  // 获取当前的菜单数据
  loadData(function (err, data) {
    if (err) return notify(err.message)

    console.log(data)
  })

  // 阻止默认事件
  return false
})
```

2.2. 保存数据逻辑

封装保存数据函数：

```
/**
 * 保存导航菜单数据
 * @param  {Array}   data      需要保存的数据
 * @param  {Function} callback 保存后需要执行的逻辑
 */
function saveData (data, callback) {
  $.post('/admin/options.php', { key: 'nav_menus', value: JSON.stringify(data) }, function (res) {
    if (!res.success) {
      return callback(new Error(res.message))
    }

    // 成功
    callback(null)
  })
}
```

实现保存逻辑:

```
/**
 * 新增逻辑
 */
$('.btn-save').on('click', function () {
  var menu = {
    icon: $('#icon').val(),
    text: $('#text').val(),
    title: $('#title').val(),
    link: $('#link').val()
  }

  // 数据校验
  for (var key in menu) {
    if (menu[key]) continue
    notify('完整填写表单')
    return false
  }

  // 获取当前的菜单数据
  loadData(function (err, data) {
    if (err) return notify(err.message)

    // 将界面上的数据追加到已有数据中
    data.push(menu)

    // 保存数据到服务端
    saveData(data, function (err) {
      if (err) return notify(err.message)
      // 再次加载
      loadData(function (err, data) {
        if (err) return notify(err.message)
        // 使用 jsrender 渲染数据到表格
        $('tbody').html($('#menu_tmpl').render(data))

        // 清空表单
        $('#icon').val('')
        $('#text').val('')
        $('#title').val('')
        $('#link').val('')
      })
    })
  })

  // 阻止默认事件
  return false
})
```

3. 删除导航菜单

3.1. 绑定删除按钮事件

将模板中每一个删除按钮调整为：

```
<a class="btn btn-danger btn-xs btn-delete" href="javascript:;" data-index="{{: #index }}">删除
</a>
```

为所有 `btn-delete` 添加点击事件：

```
// 删除按钮是后续创建的所以不能直接绑定事件，这里使用委托事件
$('tbody').on('click', '.btn-delete', function () {
  // TODO: ...
})
```

思路也是获取已有数据，在已有数据中找到当前数据并移除

```
/**
 * 删除指定数据
 */
$('tbody').on('click', '.btn-delete', function () {
  var index = parseInt($(this).parent().parent().data('index'))

  // 获取当前的菜单数据
  loadData(function (err, data) {
    if (err) return notify(err.message)

    data.splice(index, 1)

    // 保存数据到服务端
    saveData(data, function (err) {
      if (err) return notify(err.message)
      // 再次加载
      loadData(function (err, data) {
        if (err) return notify(err.message)
        $('tbody').html($('#menu_tmpl').render(data))
      })
    })
  })
})
```

> 思考：这样处理是否会有问题，为什么，如何解决