Урок N8

WebSockets

Для создания приложений с функциями обмена данными в реальном времени (чатов, игр и т.п.) прекрасно подходит технология WebSockets. Вебсокеты позволяют создать прямое сетевое соединение для быстрого и удобного двустороннего обмена данными между двумя программами.

В отличие от AJAX, после установки соединения данный метод позволяет инициировать отправку данных не только со стороны браузерного клиента на сервер, но и со стороны сервера в любое время, без каких-либо ограничений и без накладных расходов на постоянное создание и закрытие соединений. За счет этого достигается максимальная скорость обмена данными.

Содержание урока

- Модуль sockets.io
- Модуль faye

Для стыковки с помощью WebSockets браузерных клиентов и клиентов на node.js удобно использовать модуль socket.io

Он реализует кроссбраузерную поддержку WebSockets в браузере в виде клиентского скрипта и поддержку на сервере в виде npm модуля socket.io

Подключение к серверу на клиенте, отправка и обработка сообщений:

```
<script src="/socket.io/socket.io.js"></script>
<script>
	var socket = io.connect('http://localhost');
	socket.on('news', function (data) {
		console.log(data);
		socket.emit('my other event', { my: 'data' });
	});
</script>
```

Обработка входящих подключений и сообщений на сервере:

```
var io = require('socket.io').listen(80);

io.sockets.on('connection', function (socket) {
	socket.emit('news', { hello: 'world' });
	socket.on('my other event', function (data) {
		console.log(data);
	});
});
```

Недостатком обычной модели использования сокетов является то, что они предназначены для обмена данными между двумя клиентами. Это неудобно, если необходимо массово оповещать различных клиентов о различных событиях (например, объявлениях в чате, новостях и т.п.)

Помимо модели взаимодействия один-к-одному (клиент и сервер) существует модель pub/sub. В ней множество клиентов может подключаться к одному серверу, подписываться на множество каналов сообщений и отправлять сообщения в разные каналы. При отправке сообщения в какой-то канал оно будет автоматически доставлено всем подписчикам этого канала.

Сервер в таком случае является связующим звеном между клиентами: выполняет сервисные функции, контролирует права доступа клиентов, маршрутизирует сообщения между каналами и реализует обработку тех или иных событий, поступающих от пользователей.

Для реализации pub/sub на основе протокола Bayeux с использованием WebSockets удобно использовать npm модуль faye.

Простейший pub/sub сервер:

```
var http = require('http'),
   faye = require('faye');

var server = http.createServer(),
   bayeux = new faye.NodeAdapter({mount: '/'});

bayeux.attach(server);
server.listen(8000);
```

Браузерный клиент:

```
var client = new Faye.Client('http://localhost:8000/');
client.subscribe('/messages', function(message) {
        alert('Got a message: ' + message.text);
});

client.publish('/messages', {text: 'Hello world'});
```

Самоконтроль

✓ Назначение WebSockets
✓ Использование socket.io
✓ pub/sub и faye

Домашнее задание

1) С помощью созданной в предыдущем уроке модели работы с объектами на сервере создайте REST API для просмотра данных с помощью метода GET.

2) Добавьте в приложение возможность импорта данных с помощью POST, модификации данных с помощью PUT и их удаления с помощью DELETE.