Single-Page App

6.170 Recitation 8

APIs

1. Decoupling of server vs. client
2. Decoupling of data vs. presentation
3. Cross-platform compatibility
4. Less data transferred
5. Allow for asynchronous user actions -- no page refresh!

TODAY: NO PAGE REFRESH!

Review: AJAX

```
$.ajax({
    url: “/freets”,
    // can also be “GET”, modern browsers support “PUT” and “DELETE”
    type: “POST”,
    data: { content: “hello!” }
}).done(function(response) {
    // do things with the response
    // if your server specifies JSON in the response,
    // the response parameter will automatically be a Javascript object
}).fail(function(err) {
    // the important thing here is err.responseText
});
```

Updating the DOM

```
$.ajax({
    url: “/freets/3/favorites”,
    type: “POST”
}).done(function(response) {
    $(‘#freet-id’).addClass(‘favorite);
});
```

Updating the DOM

```
$.ajax({
    url: “/freets”,
    type: “POST”,
    data: { content: content }
}).done(function(response) {
    var newfreet = ‘<div class=”freet”><span class=”author”>’ +
response.author + ‘</span><p class=”freet-content”>’ + response.content
+ ‘</p></div>’;
    $(‘#freet-list’).append(newfreet);
});
```

THAT WAS UGLY

I want my views and `res.render()`

More cleanly with jQuery

```
var renderFreet = function(author, content) {
    var authorElt = $(‘<span />’).addClass(‘author’).text(author);
    var contentElt = $(‘<p />’).addClass(‘freet-content’).text(content);
    return $(‘<div />’)
        .addClass(‘freet’).append(authorElt).append(contentElt);
};

//AJAX call
...
.done(function(response) {
    $(‘#freet-list’).append(renderFreet(response.author, response.
content));
});
```

Almost a template ...

```
<div class=”freet hidden” id=”freet-template”>
    <span class=”author”></span>
    <p class=”freet-content”></p>
</div>
```

Almost a template ...

```
var renderFreet = function(author, content) {
    var freetElt = $(‘#freet-template’).clone().removeClass(‘hidden’);
    freetElt.find(‘.author’).text(author);
    freetElt.find(‘.freet-content’).text(content);
    return freetElt;
};

// AJAX call
…
.done(function(response) {
    $(‘#freet-list’).append(renderFreet(response.author, response.
content);
});
```

Client-side templates!

1. Install: `$ npm install -g handlebars`
2. Write templates in a templates/ directory with .handlebars extension
3. Compile:
   ```
   $ handlebars templates/ > public/javascripts/templates.js
   ```
4. Download Handlebars runtime, include in scripts
5. Call `Handlebars.templates[‘templateName’](data)` in your JS

Handlebars template example

```
<div class=”freet”>
    {{#if favorite}}
        <span class=”favorite”>Favorite!</span>&nbsp;
    {{/if}}
    <span>Posted by {{author}} on {{createdTime}}</span>
    <p>{{contents}}</p>
</div>
```

Partials

```
Handlebars.registerPartial(‘freet’, Handlebars.templates[‘freet’]);

    {{#each freets}}
        {{ > freet }}
    {{/each}}
```

Even more MVC?