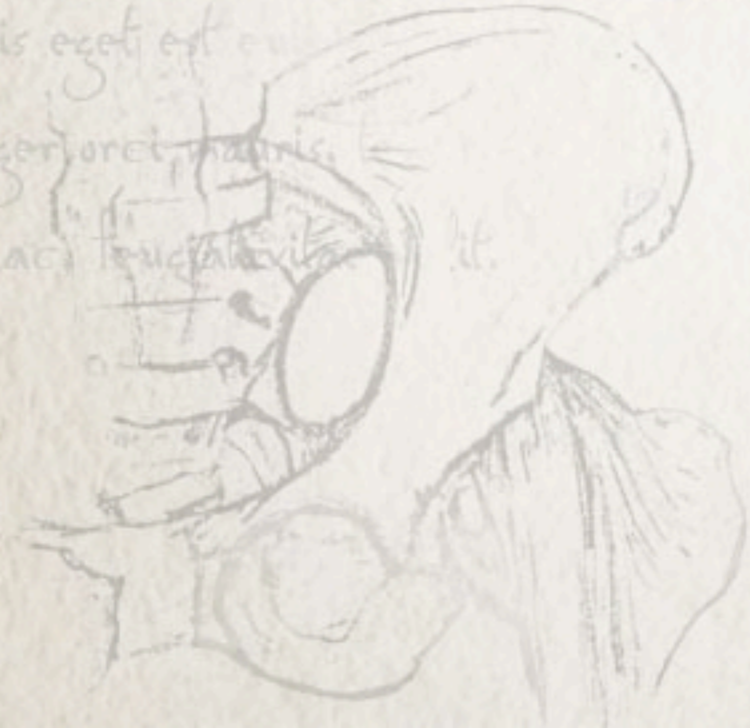
Varying Views

- LEVEL 4 -



View Initialization Options Review

Pass in the model

```
var todoView = new TodoView({
  model: todoItem
});
```

```
todoView.model == todoItem
```

Or the collection

```
var todoView = new TodoView({
  collection: todoItems
});
```

```
todoView.collection == todoItems
```

What other options can you pass into View initialization?

Using Existing DOM Elements

```
var TodoView = Backbone.View.extend({
  template: _.template("<%= description %>"),
  render: function(){
    this.$el.html(this.template(this.model.attributes));
    return this;
  }
});
```

```
var todoView = new TodoView({model: todoItem});
```

```
<div class="todo">
</div>
```

```
$('.todo').html(todoView.render().el);
```

--->

```
<div class="todo">
  <div>Pickup Milk.</div>
</div>
```

Using Existing DOM Elements

```
var TodoView = Backbone.View.extend({
  template: _.template("<%= description %>"),
  render: function(){
    this.$el.html(this.template(this.model.attributes));
    return this;
  }
});
```

```
var todoView = new TodoView({model: todoItem, el: $('.todo')});
```

```
<div class="todo">
</div>
```

```
todoView.render()
```

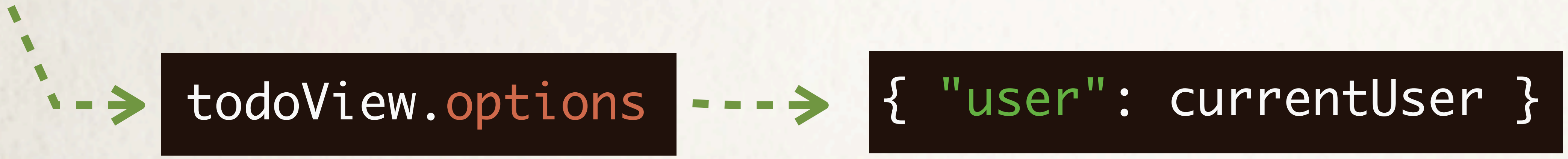
--->

```
<div class="todo">
   Pickup Milk.
</div>
```

Custom Initialization Options

Pass in an extra option

```
var todoView = new TodoView({
  model: todoItem,
  user: currentUser
});
```



```
todoView.options
```

```
{ "user": currentUser }
```

Access extra options in initialize

```
var TodoView = Backbone.View.extend({
  initialize: function(options){
    this.user = options.user;
  }
});
```

Escaping User Content

Rendering user supplied strings can lead to XSS attack

```
var TodoView = Backbone.View.extend({
  template: _.template('<%= description %>'),
  render: function(){
    this.$el.html(this.template(this.model.attributes));
    return this;
  }
});
```

```
var todoView = new TodoView({model: todoItem});
```

```
todoItem.set('description', "<script src='attack.js' />")
```

```
todoView.render().el
```

```
<div>
  <script src='attack.js' />
</div>
```

Escaping User Content

Rendering user supplied strings can lead to XSS attack

```
var TodoView = Backbone.View.extend({
  template: _.template('<%= model.escape("description") %>'),
  render: function(){
    this.$el.html(this.template({model: this.model}));
    return this;
  }
});
```

```
var todoView = new TodoView({model: todoItem});
```

```
todoItem.set('description', "<script src='attack.js' />")
```

```
todoView.render().el
```

```
<div>
  &lt;script src=&#x27;attack.js&#x27; /&gt;
</div>
```

✓

Passing extra options to event handlers

```
var TodoView = Backbone.View.extend({
  template: _.template('<span><%= description %></span>'),
  initialize: function(){
    this.model.on('change:description', this.change, this);
  },
  render: function(){
    this.$el.html(this.template(this.model.attributes));
  },
  change: function(model, value){
    this.$('span').html(value);
    this.$el.effect("highlight", {}, 1000);
  },
});
```

```
todoItem.set({description: "Pickup Kids"});
```

---> 'change:description'

How do make a change without firing off events?

Passing extra options to event handlers

```
var TodoView = Backbone.View.extend({
  template: _.template('<span><%= description %></span>'),
  initialize: function(){
    this.model.on('change:description', this.change, this);
  },
  render: function(){
    this.$el.html(this.template(this.model.attributes));
  },
  change: function(model, value){
    this.$('span').html(value);
    this.$el.effect("highlight", {}, 1000);
  },
});
```

```
todoItem.set({description: "Pickup Kids"}, {silent: true});
```

How do we stop just the highlight from happening?

Passing extra options to event handlers

```
var TodoView = Backbone.View.extend({
  template: _.template('<span><%= description %></span>'),
  initialize: function(){
    this.model.on('change:description', this.change, this);
  },
  render: function(){
    this.$el.html(this.template(this.model.attributes));
  },
  change: function(model, value, options){
    this.$('span').html(value);
    if (options.highlight !== false){
      this.$el.effect("highlight", {}, 1000);
    }
  },
});
```

```
todoItem.set({description: "Pickup Kids"}, {highlight: false});
```

✓

View Event Cleanup

```
var TodoView = Backbone.View.extend({
  initialize: function() {
    this.model.on('change', this.render, this);
  }
});
```

```
todoView.remove();
```

✗ **Model still holds reference to view**

```
var TodoView = Backbone.View.extend({
  remove: function() {
    Backbone.View.prototype.remove.apply(this, arguments);
    this.model.off(null, null, this);
  }
});
```

Old Backbone

View Event Cleanup

```
var TodoView = Backbone.View.extend({
  initialize: function() {
    this.listenTo(this.model, 'change', this.render);
  }
});
```

View "listens to" a model ✓

Stop all listeners for this view

```
todoView.stopListening();
```

Will automatically call stopListening()

```
todoView.remove();
```

Added in Backbone 0.9.9

Varying Views

- LEVEL 4 -