Marionette.Renderer

Объект `Renderer` был извлечен из процесса рендеринга `ItemView`, чтобы обеспечить удобный и отчетливый метод рендеринга шаблона с данными или без них.

Базовое использование

Базовое использование `Renderer`-а сводится к вызову метода `render`.
Этот метод возвращает строку, содержащую результат обработки шаблона в контексте объекта `data`.

```
var template = "#some-template";
var data = {foo: "bar"};
var html = Backbone.Marionette.Renderer.render(template, data);
// делаем что-то с HTML
```

Прекомпиляция шаблонов

Если параметр `data` функции `render` сам является функцией, renderer считает, что имеет дело с заранее компилированным шаблоном, и не пытается откомпилировать его еще раз. Благодаря этому любое представление,
в котором задан параметр `template`, будет брать в качестве его значения функцию, представляющую собой откомпилированный шаблон.

```
var myTemplate = _.template("<div>foo</div>");
  Backbone.Marionette.ItemView.extend({
  template: myTemplate
});
```

Что касается функции, возвращающей откомпилированный шаблон, то нужно заметить, что ее выбор не накладывает никаких ограничений - она может быть взята из api любого движка шаблонов.
Важно только, чтобы она действительно возвращала валидный HTML в качестве строки из переданных в нее данных.

Пользовательский механизм загрузки и рендеринга шаблонов

По умолчанию, renderer принимает jQuery-селектор в качестве первого параметра, и JSON-объект в качестве опционального второго параметра.
Почле чего он загружает шаблон из `Marionette.TemplateCache`, пользуясь в качестве ключа указанным селектором, и рендерит в шаблон предоставленные в качестве параметра данные, если таковые имеются, с помощью Underscore.js микро-шаблонизатора.

Если есть необходимость переопределить способ, которым загружается шаблон, вам следует обратить внимание на объект

`TemplateCache`.

Если же вы хотите назначить другой движок шаблонов, вместо используемого по умолчанию, переопределите метод `render`:

```
Backbone.Marionette.Renderer.render = function(template, data){
  return $(template).tmpl(data);
});
```

В этом примере движок шаблонов Underscore.js заменен на аналогичный из jQuery.

Если вы переопредяете метод `render`, но, тем не менее, хотите продолжать использовать механизм `TemplateCache`, не забудьте вставить строчку кода, отвечающую за загрузку шаблона из кеша:

```
Backbone.Marionette.Renderer.render = function(template, data){
var template = Marionette.TemplateCache.get(template);
// делаем что-то с шаблоном...
};
```

Использование прекомпилированных шаблонов

Стандартный механизм рендеринга шаблонов легко заменяется с помощью предварительно скомпилированных шаблонов, если используются плагины JST или TPL для AMD/RequireJS.

(см. также дополнение к документации - "Using marionette with requirejs")

Просто переопределите метод `render`, чтобы он возвращал необходимую функцию, с переданными в нее данными:

```
Backbone.Marionette.Renderer.render = function(template, data){
   return template(data);
});
```

После этого вы можете использовать прекомпилированную функцию для работы с шаблоном в качестве значения атрибута `template`:

```
var myPrecompiledTemplate = _.template("<div>some template</div>");
Backbone.Marionette.ItemView.extend({
  template: myPrecompiledTemplate
});
```