模板

什么是模板引擎

模板引擎（这里特指用于Web开发的模板引擎）是为了使用户界面与业务数据（内容）分离而产生的，它可以生成特定格式的文档，用于网站的模板引擎就会生成一个标准的HTML文档。 模板引擎不属于特定技术领域，它是跨领域跨平台的概念。在Asp下有模板引擎，在PHP下也有模板引擎，在C#下也有，甚至JavaScript、WinForm开发都会用到模板引擎技术。

为什么需要用模板引擎

我们在做前端开发的时候，有时候经常需要根据后端返回的json数据，然后来生成html，再显示到页面中去。

```
var data = [
    {text: "测试一"},
    {text: "测试二"},
    {text: "测试三"},
    {text: "测试四"}
];
function generateList(data) {
    var listHtml = "";
    listHtml += "<ul>";
    for (var i = 0, len = data.length; i < len; i++) {
        listHtml += "<li>";
        listHtml += data.text;
        listHtml += "</li>";
```

```
    }
    listHtml += "</ul>";
    return listHtml;
}
```

但是，这种通过字符串拼接的方式，比较简单的还好，如果结构比较复杂，拼接的时候还需要注意引号之间的嵌套，这样的代码维护起来比较困难。

一旦需求发生变化，这里修改起来也是很麻烦。所以我们需要模板引擎来改善这种情况。

例如上面的例子，如果使用模板引擎则可以是这样子：

```
var data = {
    list:[
        {text: "测试一"},
        {text: "测试二"},
        {text: "测试三"},
        {text: "测试四"}
    ]
};
<script id="test" type="text/html">
    <ul>
        <% for (var i = 0; i < list.length; i ++) { %>
        <li><%= list[i].text %></li>
        <% } %>
    </ul>
</script>
```

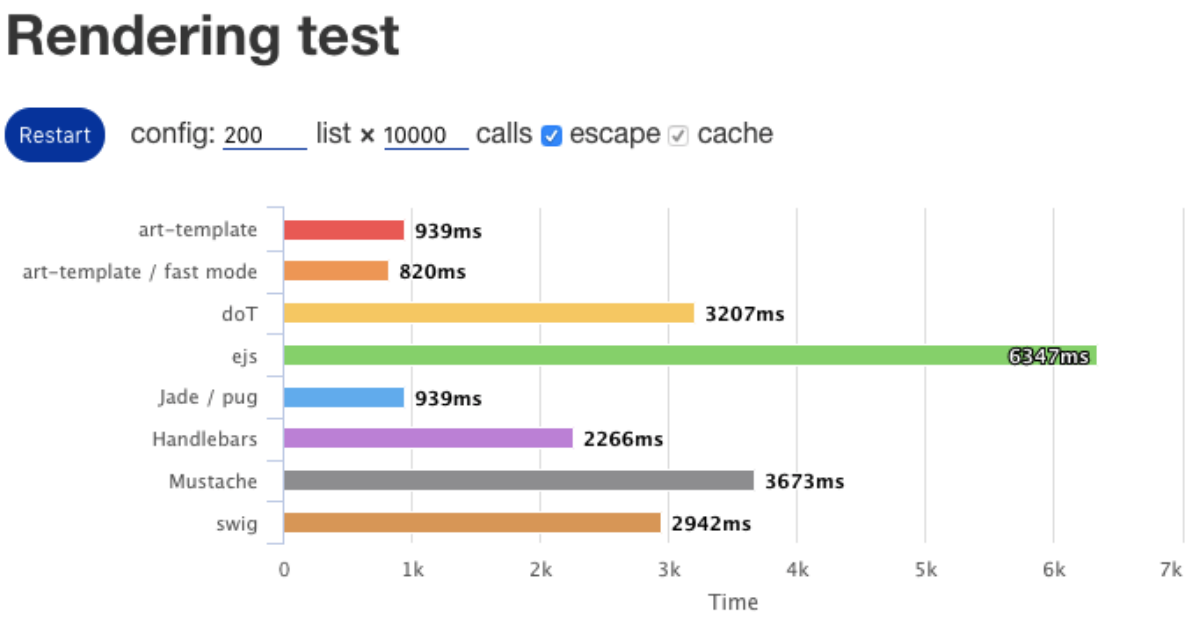
模拟模板引擎

要求：

```
    //模板
    var str = "Hello, {{name}} , 年龄 {{age}}";
    //数据
    var obj = { name: "byesame",age:20 };
    //使用方法将str变为
    //"Hello, byesame , 年龄 20"
```

常见模板引擎及性能测试

artTemplate doT ejs Jade / pug Handlebars Mustache swig



art-template

art-template 是一个简约、超快的模板引擎。 它采用作用域预声明的技术来优化模板渲染速度，从而获得接近 JavaScript 极限的运行性能，并且同时支持 NodeJS 和浏览器。

在浏览器使用

兼容:IE8+（IE8 需要补丁才能运行。）

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>IE</title>

<!--<script src="https://cdnjs.cloudflare.com/ajax/libs/es5-shim/4.5.7/es5-
shim.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/es5-shim/4.5.7/es5-
sham.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/json3/3.3.2/json3.min.js">
</script>-->
<script src="es5-shim.min.js"></script>
<script src="es5-sham.min.js"></script>
<script src="json3.min.js"></script>

<script src="../../lib/template-web.js"></script>
</head>

<body>
<div id="content"></div>
<!--这个标签里面是写好的模板-->
```

```
<script id="test" type="text/html">
{{if isAdmin}}

<h1>{{title}}</h1>
<ul>
    {{each list value i}}
        <li>索引 {{i + 1}} ：{{value}}</li>
    {{/each}}
</ul>

{{/if}}
{{$data}}
</script>

<script>
//数据
var data = {
    title: '基本例子',
    isAdmin: true,
    list: ['文艺', '博客', '摄影', '电影', '民谣', '旅行', '吉他']
};
//通过引入art-template里面的template方法 传入 模板和数据 返回 生成好的html代码
var html = template('test', data);
document.getElementById('content').innerHTML = html;
</script>
</body>
</html>
```

在node中使用

```
var template = require('art-template');
var html = template(__dirname + '/tpl-user.art', {
    user: {
        name: 'aui'
    }
});
```

在express中使用

npm install --save art-template npm install --save express-art-template

```
var express = require('express');
var app = express();
app.engine('art', require('express-art-template'));
app.set('view options', {
    debug: process.env.NODE_ENV !== 'production'
});
```

```
app.get('/', function (req, res) {
    res.render('index.art', {
        user: {
            name: 'aui',
            tags: ['art', 'template', 'nodejs']
        }
    });
});
```

语法

输出

```
{{value}}
{{data.key}}
{{data['key']}}
{{a ? b : c}}
{{a || b}}
{{a + b}}
```

原文输出

```
{{@ value}}
```

条件

```
{{if value}} ... {{/if}}
{{if v1}} ... {{else if v2}} ... {{/if}}
```

循环

```
{{each target}}
    {{$index}} {{$value}}
{{/each}}
```

1. target 支持 array 与 object 的迭代，其默认值为 $data。
2. $value$与index 可以自定义：{{each target val key}}。

模板继承

```
{{extend './layout.art'}}
{{block 'head'}} ... {{/block}}
```

模板继承允许你构建一个包含你站点共同元素的基本模板“骨架”。范例：

```
<!--layout.art-->
<!doctype html>
<html>
<head>
    <meta charset="utf-8">
    <title>{{block 'title'}}My Site{{/block}}</title>

    {{block 'head'}}
    <link rel="stylesheet" href="main.css">
    {{/block}}
</head>
<body>
    {{block 'content'}}{{/block}}
</body>
</html>
```

```
<!--index.art-->
{{extend './layout.art'}}

{{block 'title'}}{{title}}{{/block}}

{{block 'head'}}
    <link rel="stylesheet" href="custom.css">
{{/block}}

{{block 'content'}}
    <p>This is just an awesome page.</p>
{{/block}}
```