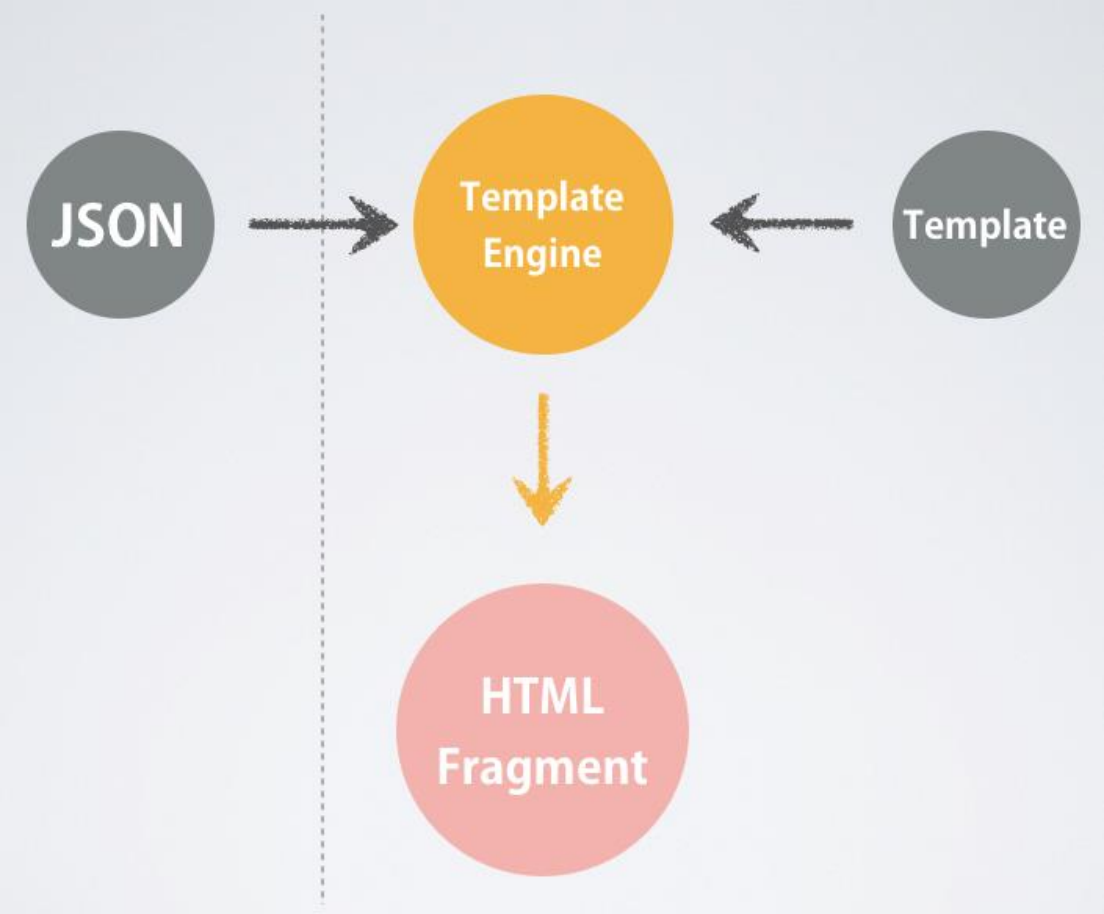
Template Engine

FE implementation & artTemplate

@Jiavan

2017/08/28

Template engine?



模板[mú bǎn]

Why use template?

低效率

```
var html = '';
for (var i = 0, users = data.users; i < users.length; i ++) {
        html += '<li><a href="'
        + users[i].url
        + '">'
        + users[i].name
        + '</a></li>';
}
//...
```

Template

- ✓ 解耦，模块化组织
- ✓ 性能，预编译
- ✓ 效率，模块复用
- ✓ 友好，简洁语法可读性强
- ✓ 工程，配合构建工具

Popular template

- ✓ pug(jade) => github 15K star
- ✓ mustache => github 2.4K star / Logic-less Ruby templates. MIUI基础应用运营系统
- ✓ ejs => github 3.5K star
- ✓ handlebars => github 12k star. 小米生活
- ✓ doT => github 3.6K star. MIUI浏览器
- ✓ artTemplate => github 5.5K start, Tencent inc. 小米生活
- ✓ velocity => Apache, http://velocity.apache.org/. MIUI基础应用运营系统
- ✓ ECMAScript 2015+ => `template syntax`.

Core implementation

文本访问 JavaScript 解析引擎

- ✓ Function
- ✓ eval
- ✓ setTimeout
- ✓ setInterval

使用Function构造器生成的Function对象是在函数创建时解析的。这比你使用函数声明或者函数表达式(function)并在你的代码中调用更为低效，因为使用后者创建的函数是跟其他代码一起解析的。

Core implementation

虽然每个引擎从模板语法、语法解析、变量赋值、字符串拼接的实现方式各有所不同，但关键的渲染原理仍然是**动态执行 javascript 字符串。**

```
<script id="test" type="text/html">
    <% if (name) { %>
        <div><%=name%></div>
    <% } %>
</script>
```

var data = { name: ‘jiavan’ };

=>

```
function render() {
    'use strict';
    var name = $data.name, $out = '';
    if (name) {
        $out += '\n<div>';
        $out += $escape(name);
        $out += '</div>\n        ';
    }
    return $out;
}
```

Core implementation

虽然每个引擎从模板语法、语法解析、变量赋值、字符串拼接的实现方式各有所不同，但关键的渲染原理仍然是**动态执行 javascript 字符串。**

```
1 var tpl = `<div><%=name%></div>`;
2 var openTag = '<%';
3 var closeTag = '%>';
4
5 var strings = tpl.split(openTag);
6 console.log(strings);
7 strings.forEach(function(code) {
8     console.log(code.split(closeTag));
9 });
```

tokenize
=>

```
▶ (2) ["<div>", "=name%></div>"]
▶ ["<div>"]
▶ (2) ["=name", "</div>"]
```

artTemplate

0 trim 方法判断 engine (IE 8 以下不支持 trim)

1 扫描模板分离 html 与 logic

2 处理 logic code 进行语法分析，提取变量至 headerCode(code.replace(/^=[#=]|[\s;]*$/g, '')) 以及 sandbox(escape “, <, >, ‘, &) 防止 XSS

3 拼接 html code 与 logic code 形成 mainCode

4 拼接 headerCode(use strict), mainCode, footerCode(return string) => code

5 返回渲染函数 new Function(‘args’, code)

artTemplate

语法分析

变量替换一般可以使用 with 来改变作用域

artTemplate 使用正则提取变量进行预编译

预编译

构建阶段完成编译，返回渲染函数，前端脱离compiler直接运行

Cache

缓存编译模板

Add or push ?

更快的字符串相加方式

很多人误以为数组 push 方法拼接字符串会比 += 快，要知道这仅仅是 IE6-8 的浏览器下。实测表明现代浏览器使用 += 会比数组 push 方法快，而在 v8 引擎中，使用 += 方式比数组拼接快 4.7 倍。所以 artTemplate 根据 javascript 引擎特性采用了两种不同的字符串拼接方式。

Catch exception

渲染错误

扫描和拼接模板时在换行时记录行号

只能捕获模板数据报错，执行过程中抛出

编译错误

JavaScript 引擎解析字符串出错，无法捕获，如使用非法嵌套不合法语法，无法定位错误，自行判断

Referencees

https://github.com/aui/art-template

https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Global_Objects/Function

Thanks for watching