1. nunjucks

Nunjucks是Mozilla开发的一个纯JavaScript编写的模板引擎，既可以用在Node环境下，又可以运行在浏览器端

2. 安装

```
npm install nunjucks
```

3. 使用

```
let nunjucks=require('nunjucks');
nunjucks.configure({autoescape: true});
let ret = nunjucks.renderString('hello {{username}}',{username: 'zfpx'});
console.log(ret);
```

- views是一个文件夹
- view目录必须创建在根目录下

```
let nunjucks=require('nunjucks');
nunjucks.configure('views',{autoescape:true});
let ret2 = nunjucks.render('index.html',{name: 'zfpx'});
console.log(ret2);
```

views\index.html

```
hello {{username}}
```

```
let express=require('express');
const nunjucks=require('nunjucks');
const path=require('path');
let app=express();
nunjucks.configure(path.resolve(__dirname,'views'),{
    autoescape: true,
    express:app
});
app.get('/',function (req,res) {
    res.render('index.html',{name:'zfpx'});
});
app.listen(8080);
```

4.语法

变量会从模板上下文获取，如果你想显示一个变量可以：

```
{{ username }}
```

过滤器是一些可以执行变量的函数，通过管道操作符 (|) 调用，并可接受参数。

```
let nunjucks=require('nunjucks');
nunjucks.configure({autoescape: true});
let ret=nunjucks.renderString(`
{{ names | join(",") }}
`,{names: ['name1','name2']});
console.log(ret);

let ret2=nunjucks.renderString(`
{{word| replace("world", "there") | capitalize}}
`,{word:'hello world'});
console.log(ret2);
```

if 为分支语句，与 javascript 中的 if 类似。

```
let nunjucks=require('nunjucks');
nunjucks.configure({autoescape: true});
let ret=nunjucks.renderString(`
{% if score > 90 %}
 优
{% elseif score>80 %}
 良
{% elseif score>70 %}
 中
{% elseif score >60 %}
 及格
{% else %}
 不及格
{% endif %}
`,{score:79});
console.log(ret);
```

for 可以遍历数组 (arrays) 和对象 (dictionaries)。

```
let nunjucks=require('nunjucks');
nunjucks.configure({autoescape: true});
let ret=nunjucks.renderString(`

 {% for item in items %}
   {{loop.index}} {{item.id}}:{{item.name}}
   {% endfor %}

`,{items: [{id:1,name:'zfpx1'},{id:2,name:'zfpx2'}]});
console.log(ret);
```

- loop.index: 当前循环数 (1 indexed)

- loop.index0: 当前循环数 (0 indexed)

- loop.revindex: 当前循环数，从后往前 (1 indexed)

- loop.revindex0: 当前循环数，从后往前 (0 based)

- loop.first: 是否第一个

- loop.last: 是否最后一个

- loop.length: 总数

- 模板继承可以达到模板复用的效果，当写一个模板的时候可以定义 "blocks"，子模板可以覆盖他

- 同时支持多层继承。**4.5.1 index.js**

```
let nunjucks=require('nunjucks');
const path=require('path');
nunjucks.configure(path.resolve(__dirname,'views'),{autoescape:true});
let ret2 = nunjucks.render('login.html',{name: 'zfpx'});
console.log(ret2);
```

4.5.2 layout.html

```
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>layout模板title>
head>
<body>
  <h1>我是头h1>
  {% block content%}
    我是layout模板的内容name= {{name}}
  {% endblock%}
  <h1>我是尾h1>
body>
html>
```

```
{% extends "layout.html" %}

{% block content%}
<form action="">
    用户名 <input type="text">
form>
{% endblock%}
```

include 可引入其他的模板，可以在多模板之间共享一些小模板，如果某个模板已使用了继承那么 include 将会非常有用。

```
let nunjucks=require('nunjucks');
const path=require('path');
nunjucks.configure(path.resolve(__dirname,'views'),{autoescape:true});
let ret2=nunjucks.render('items.html',{items: [{id:1,name:'zfpx1'},{id:2,name:'zfpx2'}]});
console.log(ret2);
```

items.html

```
{% extends "layout.html" %}

{% block content %}
    <ul>
        {% for item in items %}
        {% include "item.html" %}
      {% endfor %}
    ul>
{% endblock %}
```

item.html

```
<li>{{item.id}}:{{item.name}}li>
```

参考资料