# 1.模板过滤器

|  |
| --- |
|  |

## 示例：

## 1.把express\_pathinfo项目复制一份改名：express\_pathinfo\_filter，

|  |
| --- |
|  |

## 2.然后在views文件夹下面新建一个test.html文件，

|  |
| --- |
|  |

## 内容如下

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Filter test</title>  </head>  <body>      <H3>{{msg|Cap}}</H3>      <h4>时间格式：{{tstamp | dateFormat 'yyyy-MM-dd hh:mm:ss'}}</h4>  </body>  </html> |

## 3.在02-art-template-filter.js中添加过滤器代码

|  |
| --- |
| let express = require('express')  const { template } = require('express-art-template')  let path = require('path')  let port = 3000  let app = express()  //定义art-template过滤器  //1.首字母大写过滤器  template.defaults.imports.Cap = (data)=>{      if(data){          return data.substring(0,1).toUpperCase()+data.substring(1)      }  }  //时间戳变为"yyyy-MM-dd hh:mm:ss" 格式的时间字符串过滤器  template.defaults.imports.dateFormat = (timestamp)=>{      let date = new Date(timestamp)      console.log(date);      let year = date.getFullYear()      let month = (date.getMonth()+1).toString().padStart(2,'0')      let day = date.getDay().toString().padStart(2,'0')      let h = date.getHours().toString().padStart(2,'0')      let m = date.getMinutes().toString().padStart(2,'0')      let s = date.getSeconds().toString().padStart(2,'0')      return `${year}-${month}-${day} ${h}:${m}:${s}`  }  //处理中文乱码  app.use(function (req, res, next) {      res.setHeader('Content-Type', 'text/html;charset=utf-8');      next();  });  //配置模板引擎,4步，一个engine和3个set  app.engine('html',require('express-art-template'))  app.set('view options',{      debug:process.env.NODE\_ENV!=='production'  })  app.set('views',path.join(\_\_dirname,'views'))  app.set('view engine','html')  //app.use(express.static(\_\_dirname+'/public'))//静态服务器，根据需要配置  app.get('/',(req,res)=>{       res.end('<a href="/list">new list</a>')  })  app.get('/list',(req,res)=>{       res.render('list') //渲染列表页面  })  //测试过滤器的路由  app.get('/testfilter',(req,res)=>{      res.render('test',{msg: 'hello',tstamp:1699978584726})  })  //响应动态路由  app.get('/detail/:id/:type',(req,res)=>{      let {id,type} = req.params      res.end(`<h1>第${id}条新闻标题</h1> <hr> <h3>主题是：${type}</h3>`)  })  app.listen(port,()=>{       console.log('server is ready :http://localhost:'+port+'/');  }) |

### 效果：

|  |
| --- |
|  |

# 2.模板继承

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **标准语法**   |  | | --- | | {{extend './layout.art'}} {{block 'head'}} ... {{/block}} |   **原始语法**   |  | | --- | | <% extend('./layout.art') %> <% block('head', function(){ %> ... <% }) %> |   模板继承允许您构建包含网站公共部分的基本模板“框架”。例：   |  | | --- | | <!--layout.art--> <!doctype html> <html> <head>  <meta charset="utf-8">  <title>{{block 'title'}}My Site{{/block}}</title>   {{block 'css'}}  <link rel="stylesheet" href="main.css">  {{/block}} </head> <body>  {{block 'content'}}{{/block}} </body> </html> |  |  | | --- | | <!--index.art--> {{extend './layout.art'}}  {{block 'title'}}{{title}}{{/block}}  {{block 'css'}}  <link rel="stylesheet" href="custom.css"> {{/block}}  {{block 'content'}} <p>This is just an awesome page.</p> {{/block}} |   渲染 index.art 后，将自动应用布局骨架。 子模板 **标准语法**   |  | | --- | | {{include './header.art'}} {{include './header.art' data}} |   **原始语法**   |  | | --- | | <% include('./header.art') %> <% include('./header.art', data) %> |  1. datavalue 是默认值。标准语法不支持声明 and but 引用变量。但是，原始语法是没有限制的。$dataobjectarray 2. art-template 内置了 HTML 压缩器，请避免在子模板中写入异常的结束标签。否则，当选项打开时，标签可能会意外地“优化”。minimize |

# 示例：

## 1把express\_pathinfo\_filter项目复制一份改名：express\_pathinfo\_inherit

|  |
| --- |
|  |

## 2.为了节省磁盘空间，我把node\_modules和package.json移动到父目录：express\_basic里面，启动项目，也是可以正常运行的

## 3.在项目根目录下面新建一个public文件夹，里面新建一个css文件夹，新建两个css文件：main.css和custom.css

### main.css

|  |
| --- |
| body{      background-color: rgb(252, 242, 230);  } |

### custom.css

|  |
| --- |
| body{      background-color: bisque;  }  hr{      border: 3px dashed deeppink;  } |

## 4.在views文件夹里面新建一个layout.html作为符模板，一个index.html继承这个layout.html

### layout.html

|  |
| --- |
| <!doctype html>  <html>  <head>      <meta charset="utf-8">      <title>{{block 'title'}}My Site{{/block}}</title>      {{block 'css'}}      <link rel="stylesheet" href="/css/main.css">      {{/block}}  </head>  <body>      {{block 'content'}}{{/block}}  </body>  </html> |

### index.html

|  |
| --- |
| {{extend './layout.html'}}  {{block 'title'}} {{title}} {{/block}}  {{block 'css'}}  <link rel="stylesheet" href="/css/custom.css"> <!--这个css文件在静态资源服务器里面  -->  {{/block}}  {{block 'content'}}  <h2>欢迎来到首页</h2>  <hr>  <h3>新闻列表：<a href={{link}}>news list</a></h3>  <!-- <img src="/images/sw.jpg" alt=""> -->  {{/block}} |

### 注意：子模板只需要写和父模板不一样的内容即可，相同的内容都可以继承下来

## 5.在03-art-template-filter-inherit.js中使用静态服务器，把public目录作为静态服务器根目录，然后提交渲染index.html的代码

|  |
| --- |
| let express = require('express')  const { template } = require('express-art-template')  let path = require('path')  let port = 3500  let app = express()  app.use(express.static(\_\_dirname+'/public'))//静态服务器，根据需要配置  //定义art-template过滤器  //1.首字母大写过滤器  template.defaults.imports.Cap = (data)=>{      if(data){          return data.substring(0,1).toUpperCase()+data.substring(1)      }  }  //时间戳变为"yyyy-MM-dd hh:mm:ss" 格式的时间字符串过滤器  template.defaults.imports.dateFormat = (timestamp)=>{      let date = new Date(timestamp)      console.log(date);      let year = date.getFullYear()      let month = (date.getMonth()+1).toString().padStart(2,'0')      let day = date.getDay().toString().padStart(2,'0')      let h = date.getHours().toString().padStart(2,'0')      let m = date.getMinutes().toString().padStart(2,'0')      let s = date.getSeconds().toString().padStart(2,'0')      return `${year}-${month}-${day} ${h}:${m}:${s}`  }  //处理中文乱码  app.use(function (req, res, next) {      res.setHeader('Content-Type', 'text/html;charset=utf-8');      next();  });  //配置模板引擎,4步，一个engine和3个set  app.engine('html',require('express-art-template'))  app.set('view options',{      debug:process.env.NODE\_ENV!=='production'  })  app.set('views',path.join(\_\_dirname,'views'))  app.set('view engine','html')  app.get('/',(req,res)=>{       res.render('index',{title:"模板继承例子",link:'/list'})  })  app.get('/list',(req,res)=>{       res.render('list') //渲染列表页面  })  //测试过滤器的路由  app.get('/testfilter',(req,res)=>{      res.render('test',{msg: 'hello',tstamp:1699978584726})  })  //响应动态路由  app.get('/detail/:id/:type',(req,res)=>{      let {id,type} = req.params      res.end(`<h1>第${id}条新闻标题</h1> <hr> <h3>主题是：${type}</h3>`)  })  app.listen(port,()=>{       console.log('server is ready :http://localhost:'+port+'/');  }) |

#### 效果

|  |
| --- |
|  |
|  |
|  |