koa

Express的下一代基于Node.js的web框架

Express有自己的内置中间件，也实现了自己的路由，但是Koa只是轻量级的框架，没有自己内置的中间件和路由，只是继承了第三方中间件.

##### 1安装

npm i koa // koa需要node v7.6.0 或更高版本来支持

##### 2定义

使用app.use()注册的函数。每次客户端的请求，koa都会调用。

##### 3基本格式

async/await替代了Promise处理异步

await 修饰的如果是Promise对象：可以获取Promise中返回的内容（resolve或reject的参数），且取到值后语句才会往下执行；

//ctx：上下文，核心对象

//next：将处理的控制权转交给下一个中间件

app.use(async (ctx, next)=>{

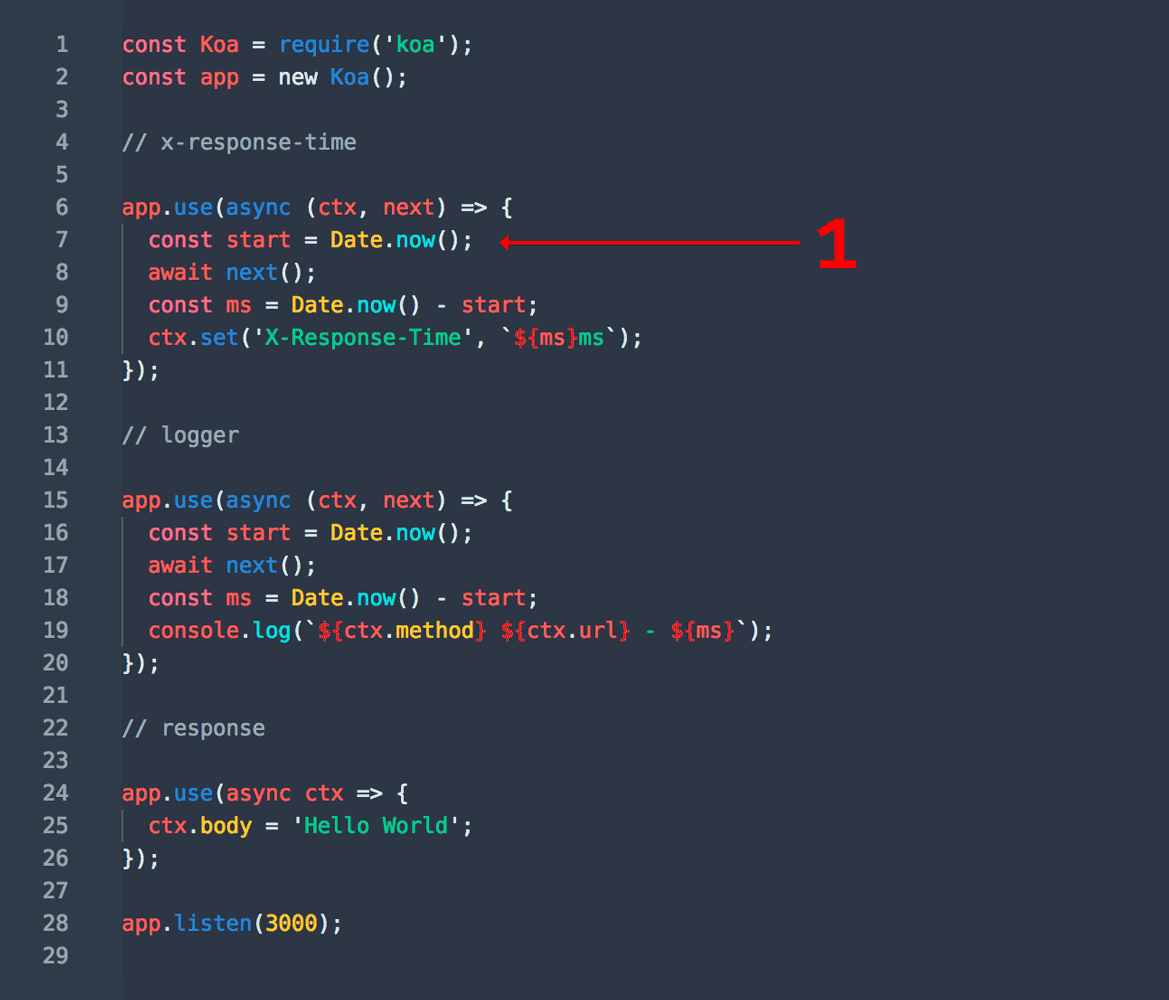
//...

await next() //等待下个中间件运行结束，才运行当前中间件的后续代码

//...

})

##### 4基本执行流程

**

### 5 koa路由

npm i koa-router

##### 6 搭建一个koa+nuxt的项目

<https://github.com/nuxt-community/koa-template> 搭建一个koa+nuxt的项目

vue init nuxt-community/koa-template <project-name>

cd <project-name> # move to your project

npm install # or yarn install\*[see note below]

##### 7 第三方中间件koa-bodyparser

koa接收到的post请求参数并不是json格式，我们需要将其转换为json，这个中间件可以将post请求的参数转为json格式返回

使用方法：安装后直接在ctx.request.body内获取POST请求参数，中间件自动给我们解析为json

const Koa = require('koa');

const app = new Koa();

// 引入koa-bodyparser中间件，这个中间件可以将post请求的参数转为json格式返回

const bodyParser = require('koa-bodyparser');

// 使用中间件后，可以用ctx.request.body进行获取POST请求参数，中间件自动给我们解析为json

app.use(bodyParser());

// request.method可以获取请求方法。get，post或者其他类型(request对象被封在ctx内，所以也可以ctx.method获取)

app.use(async(ctx)=>{

if (ctx.url === '/' &&ctx.method === 'POST'){

let postdata = ctx.request.body

ctx.body = postdata

}else {

// 其他请求显示404

ctx.body = '<h1>404!</h1>'

}

})

app.listen(3000,()=>{

console.log('server starting at localhost:3000')

})

<https://www.jianshu.com/p/ff18cee16aed>

https://www.cnblogs.com/cckui/p/9964171.html

##### 8 一些koa的插件

npm install --save koa-views

npm install ejs --save // 不太好用不推荐

npm install mongoose // 在node.js异步环境下对mongodb进行便捷操作的对象模型工具

npm install koa-bodyparser // 处理post请求

npm install koa-cors // 解决跨域

npm install --save koa-convert // 兼容koa低版本插件

npm install koa-static // 静态资源中间件 静态web服务首先去static目录匹配 ，如果能找到返回对应的文件，找不到 next()

npm install jsonwebtoken –save // JSON Web Token（缩写 JWT）是目前最流行的跨域认证解决方案