express中使用session

# 介绍

|  |
| --- |
|  |

# 案例：在express中使用session，注意需要启动mongodb数据库服务

## 在nodejs-code的session-control下面新一个02-session.js文件

## 1.需要一个插件express-session，nodejs-code目录下面安装

### 安装：npm i express-session

|  |
| --- |
|  |

## 安装connect-mongo插件,express-session默认是保存在内存中的，我们用connect-mongo把它保存到数据库

### npm i connect-mongo

|  |
| --- |
|  |

## 2.利用我们的快捷键代码判断etp生成基本代码

|  |
| --- |
| let express = require('express')  let port = 8000  let app = express()  //app.use(express.static(\_\_dirname+'/public'))//静态服务器，根据需要配置  app.get('/',(req,res)=>{       res.end('<h1>welcome</h1>')  })  app.listen(port,()=>{       console.log('server is ready :http://localhost:'+port+'/');  }) |

## 3.导入express-session和connect-mongo并且设置session

|  |
| --- |
| let express = require('express')  //导入express-session和connect-mongo  let session = require('express-session')  let MongoStore = require('connect-mongo')  let port = 8000  let app = express()  //app.use(express.static(\_\_dirname+'/public'))//静态服务器，根据需要配置  //设置并且全局注册express-session中间件  app.use(session({      name:'sid',//设置cookie的name，默认是connect.sid      secret:"atguigu",//参与加密的字符串，又叫做签名      saveUninitialized:false,//是否为每一次请求都设置一个cookie用来存储session的id      resave:true,//是否在每一次请求时重新保存session      store:MongoStore.create({          mongoUrl:'mongodb://127.0.0.1:27017/project' //数据库连接配置      }),      cookie:{          httpOnly:true, //开启后，前端无法通过js操作          maxAge:1000\*60\*5 //控制session id的过期时间和cookie的过期时间      }  }))  app.get('/',(req,res)=>{       res.end('<h1>welcome</h1>')  })  app.listen(port,()=>{       console.log('server is ready :http://localhost:'+port+'/');  }) |

## 4.下面我们用express来具体操作session

### 我们打开一个终端然后定位到02-session.js设置目录，然后运行nodemon .\02-session.js,然后我们在浏览器访问我们的服务，再打开一个cmd窗口，运行mongo，然后运行show dbs；发现多了一个project数据库，然后我们用show collections命令，发现有一个sessions集合，不过此时它是空的

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

## 5.我们添加一个login路由规则，为了方便用apipost测试，我们把端口号改为3000

|  |
| --- |
| let express = require('express')  //导入express-session和connect-mongo  let session = require('express-session')  let MongoStore = require('connect-mongo')  let port = 3000  let app = express()  //app.use(express.static(\_\_dirname+'/public'))//静态服务器，根据需要配置  //设置并且全局注册express-session中间件  app.use(session({      name:'sid',//设置cookie的name，默认是connect.sid      secret:"atguigu",//参与加密的字符串，又叫做签名      saveUninitialized:false,//是否为每一次请求都设置一个cookie用来存储session的id      resave:true,//是否在每一次请求时重新保存session      store:MongoStore.create({          mongoUrl:'mongodb://127.0.0.1:27017/project' //数据库连接配置      }),      cookie:{          httpOnly:true, //开启后，前端无法通过js操作          maxAge:1000\*60\*5 //控制session id的过期时间和cookie的过期时间      }  }))  app.get('/',(req,res)=>{       res.end('<h1>welcome</h1>')  })  //添加login路由  app.get('/login',(req,res)=>{     //username=admin&password=admin     if(req.query.username==='admin' && req.query.password==='admin'){        //设置session信息        req.session.username='admin'        req.session.uid = '258aefccc'        res.send('登录成功')     } else {      res.send('登录失败')     }  })  app.listen(port,()=>{       console.log('server is ready :http://localhost:'+port+'/');  }) |

### 我们用apipost新建一个get请求，然后测试一下

|  |
| --- |
|  |

### 此时窗口数据库，发现有内容了

|  |
| --- |
|  |

## 6.我们来读取session，我们添加一个购物车路由，他必须是登录了才能使用

|  |
| --- |
| let express = require('express')  //导入express-session和connect-mongo  let session = require('express-session')  let MongoStore = require('connect-mongo')  let port = 3000  let app = express()  //app.use(express.static(\_\_dirname+'/public'))//静态服务器，根据需要配置  //设置并且全局注册express-session中间件  app.use(session({      name:'sid',//设置cookie的name，默认是connect.sid      secret:"atguigu",//参与加密的字符串，又叫做签名      saveUninitialized:false,//是否为每一次请求都设置一个cookie用来存储session的id      resave:true,//是否在每一次请求时重新保存session      store:MongoStore.create({          mongoUrl:'mongodb://127.0.0.1:27017/project' //数据库连接配置      }),      cookie:{          httpOnly:true, //开启后，前端无法通过js操作          maxAge:1000\*60\*5 //控制session id的过期时间和cookie的过期时间      }  }))  app.get('/',(req,res)=>{       res.end('<h1>welcome</h1>')  })  //添加login路由  app.get('/login',(req,res)=>{     //username=admin&password=admin     if(req.query.username==='admin' && req.query.password==='admin'){        //设置session信息        req.session.username='admin'        req.session.uid = '258aefccc'        res.send('登录成功')     } else {      res.send('登录失败')     }  })  app.get('/cart',(req,res)=>{      //访问购物车是需要登录的      //可以检测session是否存在用户数据来查看用户是否登录      if(req.session.username){          res.send(`${req.session.username},欢迎来到购物车页面`)      }else{          res.send("你没有登录，请先登录")      }  })  app.listen(port,()=>{       console.log('server is ready :http://localhost:'+port+'/');  }) |

### 我们可以测试一下

|  |
| --- |
|  |

### 如果你没有登录，上面的路由就会返回没有登录的信息

|  |
| --- |
|  |

## 7.编写session销毁的路由，也就是logout路由

|  |
| --- |
| let express = require('express')  //导入express-session和connect-mongo  let session = require('express-session')  let MongoStore = require('connect-mongo')  let port = 3000  let app = express()  //app.use(express.static(\_\_dirname+'/public'))//静态服务器，根据需要配置  //设置并且全局注册express-session中间件  app.use(session({      name:'sid',//设置cookie的name，默认是connect.sid      secret:"atguigu",//参与加密的字符串，又叫做签名      saveUninitialized:false,//是否为每一次请求都设置一个cookie用来存储session的id      resave:true,//是否在每一次请求时重新保存session      store:MongoStore.create({          mongoUrl:'mongodb://127.0.0.1:27017/project' //数据库连接配置      }),      cookie:{          httpOnly:true, //开启后，前端无法通过js操作          maxAge:1000\*60\*5 //控制session id的过期时间和cookie的过期时间      }  }))  app.get('/',(req,res)=>{       res.end('<h1>welcome</h1>')  })  //添加login路由  app.get('/login',(req,res)=>{     //username=admin&password=admin     if(req.query.username==='admin' && req.query.password==='admin'){        //设置session信息        req.session.username='admin'        req.session.uid = '258aefccc'        res.send('登录成功')     } else {      res.send('登录失败')     }  })  app.get('/cart',(req,res)=>{      //访问购物车是需要登录的      //可以检测session是否存在用户数据来查看用户是否登录      if(req.session.username){          res.send(`${req.session.username},欢迎来到购物车页面`)      }else{          res.send("你没有登录，请先登录")      }  })  //用户推出登录服务器需要销毁session  app.get('/logout',(req,res)=>{      req.session.destroy(()=>{          res.send('退出成功...')      })  })  app.listen(port,()=>{       console.log('server is ready :http://localhost:'+port+'/');  }) |

### 测试一下，成功

|  |
| --- |
|  |

### 数据库里面的数据也没有了

|  |
| --- |
|  |