homepage 第六期：user 微服务搭建，单元测试，可用性测试准备

作者：ChenforCode
原文链接：https://ld246.com/article/1573658260520
来源网站：链滴

1.首先创建服务层接口IUserService，里边包含了三个方法，即创建用户，根据id获取用户，和根据i获取用户和用户的课程信息

```
package cn.chenforcode.homepage.service;

import cn.chenforcode.homepage.UserInfo;
import cn.chenforcode.homepage.vo.CreateUserRequest;
import cn.chenforcode.homepage.vo.UserCourseInfo;

/**
 * @author <a href="http://www.chenforcode.cn">PKUCoder</a>
 * @date 2019/11/13 2:33 下午
 * @description 用户服务的服务层接口
 */
public interface IUserService {
    /**
     * @Author <a href="http://www.chenforcode.cn">PKUCoder</a>
     * @Date 2019/11/13 2:35 下午
     * @Param [request]
     * @Return cn.chenforcode.homepage.UserInfo
     * @Description 创建一个user
     **/
    UserInfo creatUser(CreateUserRequest request);

    /**
     * @Author <a href="http://www.chenforcode.cn">PKUCoder</a>
     * @Date 2019/11/13 2:36 下午
     * @Param [id]
     * @Return cn.chenforcode.homepage.UserInfo
     * @Description 根据id获取userinfo信息
     **/
    UserInfo getUserInfo(Long id);
```

```
    /**
     * @Author <a href="http://www.chenforcode.cn">PKUCoder</a>
     * @Date 2019/11/13 2:36 下午
     * @Param [id]
     * @Return cn.chenforcode.homepage.vo.UserCourseInfo
     * @Description 获取用户的信息和课程信息
     **/
    UserCourseInfo getUserCourseInfo(Long id);
}
```

2.对该接口进行实现，这里对大部分的代码都进行了注释。但是要注意一点，在获取用户的课程信息时候是调用了课程微服务，所以需要把courseClient也注入进来。但是从userClient获取courseInfo时候注意调用函数所需要的参数。

```
package cn.chenforcode.homepage.service.impl;

import cn.chenforcode.homepage.CourseInfo;
import cn.chenforcode.homepage.CourseInfosRequest;
import cn.chenforcode.homepage.UserInfo;
import cn.chenforcode.homepage.client.CourseClient;
import cn.chenforcode.homepage.entity.HomepageUser;
import cn.chenforcode.homepage.entity.HomepageUserCourse;
import cn.chenforcode.homepage.repository.HomepageUserCourseRepository;
import cn.chenforcode.homepage.repository.HomepageUserRepository;
import cn.chenforcode.homepage.service.IUserService;
import cn.chenforcode.homepage.vo.CreateUserRequest;
import cn.chenforcode.homepage.vo.UserCourseInfo;
import lombok.extern.slf4j.Slf4j;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.util.CollectionUtils;

import java.util.Collections;
import java.util.List;
import java.util.Optional;
import java.util.stream.Collectors;

/**
 * @author <a href="http://www.chenforcode.cn">PKUCoder</a>
 * @date 2019/11/13 2:37 下午
 * @description 用户服务服务层实现类
 */
@Slf4j
@Service
public class UserServiceImpl implements IUserService {
    @Autowired
    private HomepageUserRepository userRepository;

    @Autowired
    private HomepageUserCourseRepository userCourseRepository;

    @Autowired
```

```
    private CourseClient courseClient;

    @Override
    public UserInfo creatUser(CreateUserRequest request) {
        //首先判断request是否合法
        if (!request.validate()) {
            return UserInfo.invalid();
        }
        HomepageUser oldUser = userRepository.findByUsername(request.getUsername());
        //判断是否已经存在了这个用户名的用户
        if (null == oldUser) {
            return UserInfo.invalid();
        }

        HomepageUser newUser = userRepository.save(
                new HomepageUser(request.getUsername(), request.getEmail())
        );
        return new UserInfo(newUser.getId(), newUser.getUsername(), newUser.getEmail());
    }

    @Override
    public UserInfo getUserInfo(Long id) {
        Optional<HomepageUser> user = userRepository.findById(id);
        //如果不存在这个id的用户
        if (!user.isPresent()) {
            return UserInfo.invalid();
        }
        HomepageUser curUser = user.get();
        return new UserInfo(curUser.getId(), curUser.getUsername(), curUser.getEmail());
    }

    @Override
    public UserCourseInfo getUserCourseInfo(Long id) {
        Optional<HomepageUser> user = userRepository.findById(id);
        //如果不存在这个id的用户
        if (!user.isPresent()) {
            return UserCourseInfo.invalid();
        }
        HomepageUser homepageUser = user.get();
        //得到userInfo
        UserInfo userInfo = new UserInfo(homepageUser.getId(), homepageUser.getUsername(),
homepageUser.getEmail());

        //开始拿到课程信息
        //拿到userCourse的信息，即用户和课程的一一对应信息
        List<HomepageUserCourse> homepageUserCourses = userCourseRepository.findAllByU
erId(
                homepageUser.getId()
        );
        //如果这个人没有课程，就返回一个只有用户信息和空列表的userCourseInfo
        if (CollectionUtils.isEmpty(homepageUserCourses)) {
            return new UserCourseInfo(userInfo, Collections.emptyList());
        }
        //理解一下这里，首先如果要拿到courseInfos就需要一个courseInfosRequest对象，然后这个
```

```
象里的属性是一个ids，即
        //所有的课程id的list，那么这个list就从homepageUser的list用lambda表达式拿到
        List<CourseInfo> courseInfos = courseClient.getCourseInfos(
                new CourseInfosRequest(homepageUserCourses.stream()
                        .map(HomepageUserCourse::getId)
                        .collect(Collectors.toList()))
        );
        return new UserCourseInfo(userInfo, courseInfos);
    }
}
```

3.用户服务controller层

```
package cn.chenforcode.homepage.controller;

import cn.chenforcode.homepage.UserInfo;
import cn.chenforcode.homepage.service.IUserService;
import cn.chenforcode.homepage.vo.CreateUserRequest;
import cn.chenforcode.homepage.vo.UserCourseInfo;
import com.alibaba.fastjson.JSON;
import lombok.extern.slf4j.Slf4j;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;


/**
 * @author <a href="http://www.chenforcode.cn">PKUCoder</a>
 * @date 2019/11/13 4:21 下午
 * @description 用户服务对外接口
 */
@Slf4j
@RestController
public class HomepageUserController {
    @Autowired
    private IUserService userService;

    @PostMapping("/create/user")
    public UserInfo createUser(@RequestBody CreateUserRequest request) {
        log.info("<homepage-user> : create user -> {}", JSON.toJSONString(request));
        return userService.creatUser(request);
    }

    @GetMapping("/get/user")
    public UserInfo getUser(Long id) {
        log.info("<homepage-user> : get user -> {}", id);
        return userService.getUserInfo(id);
    }

    @GetMapping("/get/user/course")
    public UserCourseInfo getUserCourseInfo(Long id) {
```

```
        log.info("<homepage-user> : get user course info -> {}", id);
        return userService.getUserCourseInfo(id);
    }
}
```

4.完成相关功能性测试，创建test resources目录，并将main目录的配置文件拷贝进来。但是由于想在测试用例中调用其他的feign接口，需要吧course服务单独启动起来，那么我们现在先不这样做，们只是测试一下user的功能。所以在配置文件中删除feign相关的配置

5.建立测试启动类

```
package cn.chenforcode.homepage;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.openfeign.EnableFeignClients;

/**
 * @author <a href="http://www.chenforcode.cn">PKUCoder</a>
 * @date 2019/11/13 4:58 下午
 * @description user服务测试启动类
 */
@EnableFeignClients(basePackages = {"cn.chenforcode.homepage"})
@SpringBootApplication
public class Application {
    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }
}
```

6.测试增加用户

```
@Test
    @Transactional
    public void testCreateUser() {
        System.out.println(userService.createUser(
                new CreateUserRequest("pkucoder", "pkucoder@qq.com")
        ));
    }
```

@Transactional是org.springframework.transaction.annotation.Transactional包下的，作用是测用例执行完之后对数据库进行回滚

6.测试获取用户信息

```
 //{"email":"pkucoder@qq.com","id":10,"username":"pkucoder"}
    @Test
    public void testGetUserInfo() {
        System.out.println(JSON.toJSONString(
                userService.getUserInfo(10L)
        ));
    }
```

7.由于测试获取课程需要先启动课程服务，在单元测试中不好实现，所以需要在后续的postman中直用http的方式去测试，所以在这里先预先加入一些mock数据，即增加homepageUserCourse

```
@Test
    public void testCreateHomepageUserCourse() {
        HomepageUserCourse userCourse1 = new HomepageUserCourse();
        userCourse1.setUserId(10L);
        userCourse1.setCourseId(8L);

        HomepageUserCourse userCourse2 = new HomepageUserCourse();
        userCourse2.setUserId(10L);
        userCourse2.setCourseId(9L);
        System.out.println(userCourseRepository.saveAll(Arrays.asList(userCourse1, userCourse2))
size());
    }
```

8.可用性测试之前的准备，配置网关服务，修改zuul的application.yml

增加如下内容，

@1.prefix的意思是所有的网关访问如下服务需要带上chenforcode的前缀，这个前缀可有可无

@2.course和user是随便起的，但是尽量和服务相关

@3.path代表所有带上这个前缀的服务都会被发送到serviceId所指定的服务中

@4.strip-prefix代表是否去除请求的前缀，置为false

```
zuul:
  prefix: /chenforcode
  routes:
    course:
      path: /homepage-course/**
      serviceId: eureka-client-homepage-course
      strip-prefix: false
    user:
      path: /homepage-user/**
      serviceId: eureka-client-homepage-user
      strip-prefix: false
```