Lamdba--- 讲师终极赛

作者：os-tll
原文链接：https://ld246.com/article/1585399310273
来源网站：链滴

PPT对应代码示例

常见大段代码

示例一

```
import java.util.ArrayList;
import java.util.List;

/**
 * @author tanglonglong  \(-_-)/
 * @version 1.0
 * @date 2020/3/9 13:58
 */
    public class Person {
        private String givenName;
        private String surName;
        private int age;
        private String sex;
        private String eMail;
        private String phone;
        private String address;

        public static class Builder{

            private String givenName="";
            private String surName="";
            private int age = 0;
            private String sex;
            private String eMail = "";
            private String phone = "";
```

```
    private String address = "";

    public Person.Builder givenName(String givenName){
        this.givenName = givenName;
        return this;
    }

    public Person.Builder surName(String surName){
        this.surName = surName;
        return this;
    }

    public Person.Builder age (int val){
        age = val;
        return this;
    }

    public Person.Builder sex(String val){
        sex = val;
        return this;
    }

    public Person.Builder email(String val){
        eMail = val;
        return this;
    }

    public Person.Builder phoneNumber(String val){
        phone = val;
        return this;
    }

    public Person.Builder address(String val){
        address = val;
        return this;
    }

    public Person build(){
        return new Person(this);
    }
}

private Person(){
    super();
}

private Person(Person.Builder builder){
    givenName = builder.givenName;
    surName = builder.surName;
    age = builder.age;
    sex = builder.sex;
    eMail = builder.eMail;
    phone = builder.phone;
    address = builder.address;
```

```
    }

    public String getGivenName(){
        return givenName;
    }

    public String getSurName(){
        return surName;
    }

    public int getAge(){
        return age;
    }
    public String getSex(){
        return sex;
    }
    public void print(){
        System.out.println(
                "\nName: " + givenName + " " + surName + "\n" +
                        "Age: " + age + "\n" +
                        "String: " + sex + "\n" +
                        "eMail: " + eMail + "\n" +
                        "Phone: " + phone + "\n" +
                        "Address: " + address + "\n"
        );
    }

    public void printName(){
        System.out.println(
                "Name: " + givenName + " " + surName);
    }

    @Override
    public String toString(){
        return "Name: " + givenName + " " + surName + "\n" + "Age: " + age + "  String: " +
ex + "\n" + "eMail: " + eMail + "\n" + "Address: " + address + "\n";
    }


    public static List<Person> createShortList(){
        List<Person> people = new ArrayList<>();

        people.add(
                new Person.Builder()
                        .givenName("Bob")
                        .surName("Baker")
                        .age(21)
                        .sex("女")
                        .email("bob.baker@example.com")
                        .phoneNumber("201-121-4678")
                        .address("44 4th St, Smallville, KS 12333")
                        .build()
        );
```

```
people.add(
        new Person.Builder()
                .givenName("Jane")
                .surName("Doe")
                .age(25)
                .sex("男")
                .email("jane.doe@example.com")
                .phoneNumber("202-123-4678")
                .address("33 3rd St, Smallville, KS 12333")
                .build()
);

people.add(
        new Person.Builder()
                .givenName("John")
                .surName("Doe")
                .age(25)
                .sex("女")
                .email("john.doe@example.com")
                .phoneNumber("202-123-4678")
                .address("33 3rd St, Smallville, KS 12333")
                .build()
);

people.add(
        new Person.Builder()
                .givenName("James")
                .surName("Johnson")
                .age(45)
                .sex("女")
                .email("james.johnson@example.com")
                .phoneNumber("333-456-1233")
                .address("201 2nd St, New York, NY 12111")
                .build()
);

people.add(
        new Person.Builder()
                .givenName("Joe")
                .surName("Bailey")
                .age(17)
                .sex("女")
                .email("joebob.bailey@example.com")
                .phoneNumber("112-111-1111")
                .address("111 1st St, Town, CA 11111")
                .build()
);

people.add(
        new Person.Builder()
                .givenName("Phil")
                .surName("Smith")
                .age(11)
```

```
                .sex("女")
                .email("phil.smith@examp;e.com")
                .phoneNumber("222-33-1234")
                .address("22 2nd St, New Park, CO 222333")
                .build()
        );

        people.add(
                new Person.Builder()
                .givenName("Betty")
                .surName("Jones")
                .age(85)
                .sex("男")
                .email("betty.jones@example.com")
                .phoneNumber("211-33-1234")
                .address("22 4th St, New Park, CO 222333")
                .build()
        );


        return people;
    }

}
```

改造前

```
/*对属性进行筛选后，进行打印*/
public class FilterPrint {

   /*筛成年男性为第一组，进行打印*/
  public void method1(List<Person> pl){
    for(Person p:pl){
      if (isManAndAdult(p)){
        printManAndAdult(p);
      }
    }
  }
   /*筛未成年女性为第二组，进行打印*/
  public void method2(List<Person> pl){
    for(Person p:pl){
      if (isWomenAndChild(p)){
        printWomenAndChild(p);
      }
    }
  }


  public boolean isManAndAdult(Person p){
    return p.getAge() >= 18  && p.getSex() == "男";
  }

  public boolean isWomenAndChild(Person p){
    return p.getAge() < 18 && p.getSex() == "女";
```

```
  }

  public void printManAndAdult(Person p){
    System.out.println("Group1 " + p.getGivenName() + " " + p.getSurName() + " age " + p.ge
Age());
  }

  public void printWomenAndChild(Person p){
    System.out.println("Group2 " + p.getGivenName() + " " + p.getSurName() + " age " + p.ge
Age());
  }
}


/*调用并打印*/
public class Main {

    public static void main(String[] args) {

        FilterPrint filterPrint = new FilterPrint();
        filterPrint.method1(Person.createShortList());
        filterPrint.method2(Person.createShortList());
    }
}
```

改造后

```
//--> 这里发生了变化   使用JAVA提供的Predicate接口<--
 public interface Predicate<T> {
   public boolean test(T t);
 }

/*对属性进行筛选后，进行打印*/
public class FilterPrint {
    /*筛成年男性为第一组，进行打印*/
    //--> 这里发生了变化  传入 Predicate 的实现 <--
    public void method1((List<Person> pl, Predicate<Person> pred){
        for(Person p:pl){
            if (pred.test(p)){
                printManAndAdult(p);
            }
        }
    }
    /*筛未成年女性为第二组，进行打印*/
    //--> 这里发生了变化  传入 Predicate 的实现 <--
    public void method2(List<Person> pl, Predicate<Person> pred){
        for(Person p:pl){
            if (pred.test(p)){
                printWomenAndChild(p);
            }
        }
    }
    public void printManAndAdult(Person p){
        System.out.println("Group1 " + p.getGivenName() + " " + p.getSurName() + " age " + p.
```

```
etAge());
    }

    public void printWomenAndChild(Person p){
        System.out.println("Group2 " + p.getGivenName() + " " + p.getSurName() + " age " + p.
etAge());
    }
}

/*调用并打印*/
public class Main{

  public static void main(String[] args){

     FilterPrint robo = new FilterPrint();
     //--> 这里发生了变化  传入 Predicate 的实现 <--
    // Predicates
    robo.method1(Person.createShortList(), p->p.getAge() >= 18  && p.getSex() == "男");
    robo.method2(Person.createShortList(),  p->p.getAge() < 18 && p.getSex() == "女" );
  }
}
```

示例二

改造前

```
/*示例二：大数据 List 集合，需要对 List 集合中的数据同标准库中数据进行对比，
  生成新增，更新，取消数据*/
import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.Callable;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import java.util.concurrent.Future;

/**
 * @author tanglonglong  \ ( -  - ) /
 * @version 1.0
 * @date 2020/3/9 13:53
 */
public class DealList1 {
    public static void main(String[] args) throws Exception {

        // 开始时间
        long start = System.currentTimeMillis();
        List<String> list = new ArrayList<String>();

        for (int i = 1; i <= 30000; i++) {
            list.add(i + "");
        }
        // 每500条数据开启一条线程
        int threadSize = 500;
        // 总数据条数
```

```
        int dataSize = list.size();
        // 线程数
        int threadNum = dataSize / threadSize + 1;
        // 定义标记,过滤threadNum为整数
        boolean special = dataSize % threadSize == 0;

        // 创建一个线程池
        ExecutorService exec = Executors.newFixedThreadPool(threadNum);
        // 定义一个任务集合
        List<Callable<Integer>> tasks = new ArrayList<Callable<Integer>>();
        Callable<Integer> task = null;
        List<String> cutList = null;

        // 确定每条线程的数据
        for (int i = 0; i < threadNum; i++) {
            if (i == threadNum - 1) {
                if (special) {
                    break;
                }
                cutList = list.subList(threadSize * i, dataSize);
            } else {
                cutList = list.subList(threadSize * i, threadSize * (i + 1));
            }
            final List<String> listStr = cutList;
            task = new Callable<Integer>() {

                @Override
                public Integer call() throws Exception {
                    System.out.println(Thread.currentThread().getName() + "线程：" + listStr);
                    //...进行数据库对比等业务代码
                    return 1;
                }
            };
            // 这里提交的任务容器列表和返回的Future列表存在顺序对应的关系
            tasks.add(task);
        }

        List<Future<Integer>> results = exec.invokeAll(tasks);

        for (Future<Integer> future : results) {
            System.out.println(future.get());
        }

        // 关闭线程池
        exec.shutdown();
        System.out.println("线程任务执行结束");
        System.err.println("执行任务消耗了 ：" + (System.currentTimeMillis() - start) + "毫秒");
    }

}
```

改造后

```
import java.util.ArrayList;
```

```
import java.util.List;

/**
 * @author tanglonglong  \ ( -  - ) /
 * @version 1.0
 * @date 2020/3/9 15:45
 */
public class DealList2 {
    public static void main(String[] args) throws Exception {

        // 开始时间
        long start = System.currentTimeMillis();
        List<String> list = new ArrayList<String>();

        for (int i = 1; i <= 3000; i++) {
            list.add(i + "");
        }
        list.parallelStream().forEach((p)->{
            //...进行数据库对比等业务代码
        });
        System.err.println("执行任务消耗了 ：" + (System.currentTimeMillis() - start) + "毫秒");
    }


}
```

决赛课件PPT.zip