

# 第一章 API

API (Application Programming Interface) , 应用程序编程接口。API是一本程序员的字典, 是JDK中提供给我们使用的类的说明文档。这些类将底层的代码实现封装了起来, 我们不需要关心这些类是如何实现的, 只要学习这些类是如何使用即可。所以我们可以通过查询API的方式, 来学习Java提供的类, 并得知如何使用他们。

## 第二章 Scanner类

Scanner 类是一个可以解析基本类型和字符串的简单文本扫描器

### 2.1 引用类型使用步骤

#### 2.1.1 导包

使用 `import` 关键字导包, 在类的所有代码之前导包, 引入要使用的类型, `java.lang` 包下的所有类无需导入

格式:

```
import 包名.类名;
```

例:

```
import java.util.Scanner;
```

#### 2.1.2 创建对象

使用该类烦人构造方法, 创见一个该类对象

格式:

```
数据类型 变量名 = new 数据类型(参数列表);
```

例:

```
scanner sc = new Scanner(System.in);
```

#### 2.1.3 调用方法

调用该类的成员方法, 完成指定功能

格式:

```
变量名.方法名();
```

例:

```
int i = sc.nextInt(); //接收一个键盘录入的整数
```

## 2.2 Scanner使用步骤

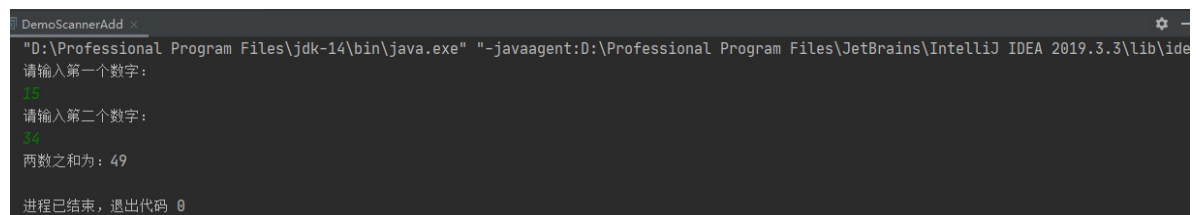
- 查看类
  - `java.util.Scanner`: 需要 `import` 导入后使用
- 查看构造方法
  - `public Scanner(InputStream source)`: 构造一个新的 `Scanner`, 它生成的值是从指定的输入流扫描的
- 查看成员方法
  - `public int nextInt()` 将输入信息的下一个标记扫描为一个 `int` 值

## 2.3 练习

### 2.3.1 求和

键盘录入两个数据并求和

```
import java.util.Scanner;
public class DemoScannerAdd {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("请输入第一个数字: ");
        int a = sc.nextInt();
        System.out.println("请输入第二个数字: ");
        int b = sc.nextInt();
        System.out.println("两数之和为: " + (a+b));
    }
}
```



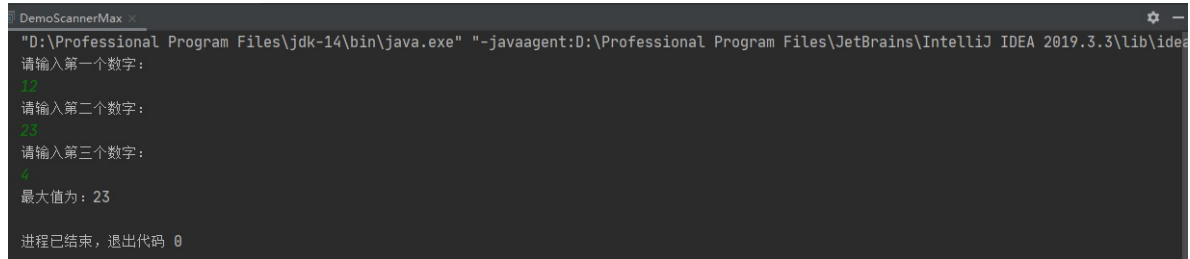
```
DemoScannerAdd x
"D:\Professional Program Files\jdk-14\bin\java.exe" "-javaagent:D:\Professional Program Files\JetBrains\IntelliJ IDEA 2019.3.3\lib\ide
请输入第一个数字:
15
请输入第二个数字:
34
两数之和为: 49
进程已结束，退出代码 0
```

### 2.3.2 取最大值

键盘录入三个数据并获取最大值

```
import java.util.Scanner;
public class DemoScannerMax {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("请输入第一个数字: ");
        int a = sc.nextInt();
        System.out.println("请输入第二个数字: ");
        int b = sc.nextInt();
        System.out.println("请输入第三个数字: ");
        int c = sc.nextInt();
        int temp = a > b ? a : b;
        int max = temp > c ? temp : c;
    }
}
```

```
        System.out.println("最大值为: " + max);  
    }  
}
```



```
DemoScannerMax  
"D:\Professional Program Files\jdk-14\bin\java.exe" "-javaagent:D:\Professional Program Files\JetBrains\IntelliJ IDEA 2019.3.3\lib\idea  
请输入第一个数字:  
12  
请输入第二个数字:  
23  
请输入第三个数字:  
4  
最大值为: 23  
进程已结束, 退出代码 0
```

## 2.4 匿名对象【了解】

创建对象时，只有创建对象语句，却没有把对象地址赋值给某个变量。虽然是创建对象的简化写法，但是应用场景非常有限

格式：

```
new 类名(参数列表);
```

### 应用场景

1. 创建匿名对象直接调用方法，没有变量名

```
new Scanner(System.in).next();
```

- 一个匿名对象，只能使用一次，一旦调用两次方法，就是创建了两个对象

2. 匿名对象可以作为方法的参数和返回值

- 作为参数

```
class Demo {  
    public static void main(String[] args) {  
        input(new Scanner(System.in));  
    }  
    public static void input(Scanner sc) {  
        System.out.println(sc);  
    }  
}
```

- 作为返回值

```
class Demo {  
    public static void main(String[] args) {  
        Scanner sc = getScanner();  
    }  
    public static Scanner getScanner() {  
        return new Scanner(System.in);  
    }  
}
```

## 第三章 Random类

此类的实例用于生成伪随机数

### 3.1 Random使用步骤

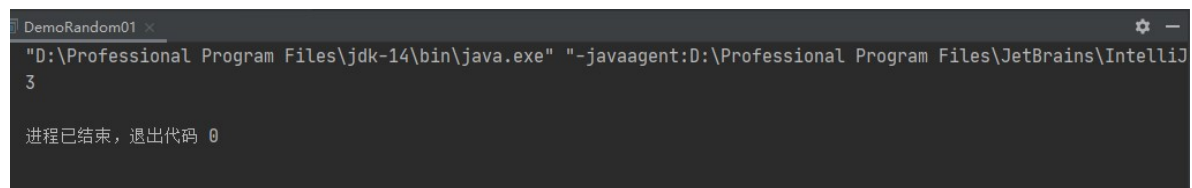
- 查看类
  - `java.util.Random`: 需要 `import` 导入后使用
- 查看构造方法
  - `public Random()`: 创建一个新的随机数生成器
- 查看成员方法
  - `public int nextInt(int n)` 返回一个伪随机数, 范围为[0,n)的 `int` 值

### 3.2 练习

#### 3.2.1 获取随机数

获取1~n之间的随机数, 包含n

```
import java.util.Random;
public class DemoRandom01 {
    public static void main(String[] args) {
        int n = 10;
        Random r = new Random();
        int number = r.nextInt(n)+1;
        System.out.println(number);
    }
}
```



DemoRandom01 x

```
"D:\Professional Program Files\jdk-14\bin\java.exe" "-javaagent:D:\Professional Program Files\JetBrains\IntelliJ"
3

进程已结束, 退出代码 0
```

#### 3.2.2 猜数字小游戏

游戏开始, 会随机生成一个1~100之间的整数, 玩家猜测一个数字, 系统提示大了或者小了, 直到玩家猜中, 游戏结束

```
import java.util.Random;
import java.util.Scanner;
public class DemoRandom02 {
    public static void main(String[] args) {
        int number = new Random().nextInt(100) + 1;
        while(true) {
            System.out.print("请输入你的猜测: ");
            int temp = new Scanner(System.in).nextInt();
            if(temp>number)
                System.out.println("大了");
            else if(temp<number)
                System.out.println("小了");
        }
    }
}
```

```
        else {  
            System.out.println("猜对了!!!");  
            break;  
        }  
    }  
}
```

```
DemoRandom02 x  
"D:\Professional Program Files\jdk-14\bin\java.exe" "-javaagent:D:\Professional Program Files\JetBrains\IntelliJ  
请输入你的猜测: 50  
小了  
请输入你的猜测: 75  
大了  
请输入你的猜测: 60  
小了  
请输入你的猜测: 65  
小了  
请输入你的猜测: 70  
小了  
请输入你的猜测: 72  
小了  
请输入你的猜测: 74  
猜对了!!!  
  
进程已结束，退出代码 0
```

## 第四章 ArrayList类

`java.util.ArrayList` 是**大小可变的数组**的实现，存储在内部的数据称为元素。此类提供一些方法来操作内部存储的元素。`ArrayList` 中可不断添加元素，其大小也自动增长

### 4.1 ArrayList 使用步骤

- 查看类
  - `java.util.ArrayList<E>`: 该类需要 `import` 导入后使用
  - `<E>`, 表示一种指定的数据类型，叫做**泛型**，取自 `Element`（元素）的首字母，在出现 `E` 的位置，我们使用一种引用数据类型将其替换即可，表示我们将存储哪种引用类型的元素
- 查看构造方法
  - `public ArrayList()`: 构造一个内容为空的集合
  - 基本格式:

```
ArrayList<String> list = new ArrayList<String>();
```

在JDK7之后，右侧泛型的尖括号之内可以留空，但是 `<>` 仍然要写。简化格式:

```
ArrayList<String> list = new ArrayList<>();
```

- 查看成员方法
  - `public boolean add(E e)`: 将指定的元素添加到此集合的尾部
  - 参数 `E e`, 在构造 `ArrayList` 对象是, `<E>` 指定了什么数据类型，那么 `add(E e)` 方法中，只能添加什么数据类型的对象

### 4.2 常用方法和遍历

- `public boolean add(E e)`: 将指定的元素添加到此集合的尾部
- `public E remove(int index)`: 移除此集合中指定位置上的元素, 返回被删除的元素
- `public E get(int index)`: 返回此集合中指定为位置上的元素, 返回获取的元素
- `public int size()`: 返回此集合中的元素数, 遍历集合时, 可以控制索引范围, 防止越界

## 4.3 如何存储基本数据类型

`ArrayList` 对象不能存储基本数据类型, 只能存储引用数据类型。但是存储基本数据类型对应的包装类型是可以的。所以, 想要存储基本类型数据, `<>` 中的数据类型, 必须经过转换后才能编写, 转换表如下:

| 基本类型    | 基本类型包装类   |
|---------|-----------|
| byte    | Byte      |
| short   | Short     |
| int     | Integer   |
| long    | Long      |
| float   | Float     |
| double  | Double    |
| char    | Character |
| boolean | Boolean   |

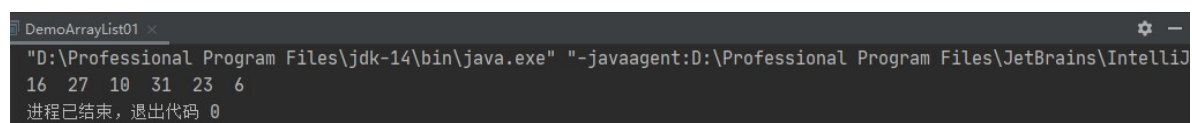
- 只有 `Integer` 和 `Character` 需要特殊记忆, 其他类型只是首字母大写即可

## 4.4 练习

### 4.4.1 数值添加到集合

生成6个1~33之间的随机整数, 添加到集合, 并遍历

```
public class DemoArrayList01 {  
    public static void main(String[] args) {  
        ArrayList<Integer> list = new ArrayList<>();  
        for (int i = 0; i < 6; i++)  
            list.add(new Random().nextInt(33) + 1);  
        for (int i = 0; i < 6; i++)  
            System.out.print(list.get(i) + "\t");  
    }  
}
```



```
DemoArrayList01 x  
"D:\Professional Program Files\jdk-14\bin\java.exe" "-javaagent:D:\Professional Program Files\JetBrains\IntelliJ  
16 27 10 31 23 6  
进程已结束, 退出代码 0
```

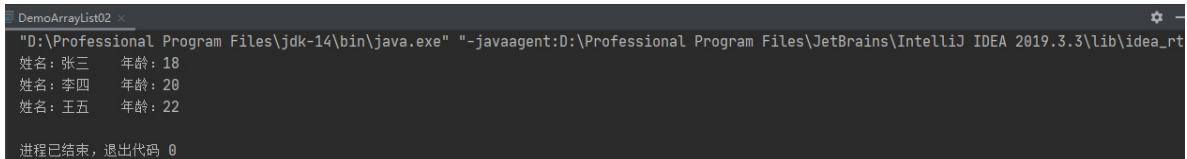
### 4.4.2 对象添加到集合

自定义3个学生对象, 添加到集合, 并遍历

```

public class DemoArrayList02 {
    public static void main(String[] args) {
        ArrayList<Student> list = new ArrayList<>();
        list.add(new Student("张三",18));
        list.add(new Student("李四",20));
        list.add(new Student("王五",22));
        for (int i = 0; i < 3; i++) {
            System.out.println("姓名: " + list.get(i).getName() + "\t年龄: "
+list.get(i).getAge());
        }
    }
}

```



```

DemoArrayList02 x
"D:\Professional Program Files\jdk-14\bin\java.exe" "-javaagent:D:\Professional Program Files\JetBrains\IntelliJ IDEA 2019.3.3\lib\idea_rt
姓名: 张三 年龄: 18
姓名: 李四 年龄: 20
姓名: 王五 年龄: 22
进程已结束，退出代码 0

```

### 4.4.3 打印集合方法

定义以指定格式打印集合的方法（`ArrayList` 类型作为参数），使用`{}`扩起集合，使用`@`分割每个元素。格式参照`{元素@元素@元素}`

```

public class DemoArrayList03 {
    public static void main(String[] args) {
        ArrayList<String> list = new ArrayList<>();
        list.add("张三");
        list.add("李四");
        list.add("王五");
        printArrayList(list);
    }
    public static void printArrayList(ArrayList<String> list) {
        System.out.print("{");
        for (int i = 0; i < list.size() - 1; i++)
            System.out.print(list.get(i) + "@");
        System.out.print(list.get(list.size()-1));
        System.out.println("}");
    }
}

```



```

DemoArrayList03 x
"D:\Professional Program Files\jdk-14\bin\java.exe" "-javaagent:D:\Professional Program Files\JetBrains\IntelliJ
{张三@李四@王五}
进程已结束，退出代码 0

```

### 4.4.4 获取集合方法

定义获取所有偶数元素集合的方法（`ArrayList` 类型作为返回值）

```

public class DemoArrayList04 {
    public static void main(String[] args) {
        ArrayList<Integer> list = new ArrayList<>();
        //向初始数列中添加20个随机数
        for (int i = 0; i < 20; i++)
            list.add(new Random().nextInt(100) + 1);
    }
}

```

```

        ArrayList<Integer> evenlist = evenList(list);
        System.out.print("初始数列: ");
        printArrayList(list);
        System.out.print("\n偶数数列: ");
        printArrayList(evenlist);
    }
    //提取数列中的偶数元素，返回一个偶数数列
    public static ArrayList<Integer> evenList(ArrayList<Integer> list) {
        ArrayList<Integer> evenlist = new ArrayList<>();
        for (int i = 0; i < list.size(); i++) {
            if (list.get(i) % 2 == 0)
                evenlist.add(list.get(i));
        }
        return evenlist;
    }
    //打印ArrayList
    public static void printArrayList(ArrayList<Integer> list) {
        for (int i = 0; i < list.size(); i++)
            System.out.print(list.get(i) + "\t");
    }
}

```

```

DemoArrayList04
"D:\Professional Program Files\jdk-14\bin\java.exe" "-javaagent:D:\Professional Program Files\JetBrains\IntelliJ IDEA 2019.3.3\lib\idea_rt.
初始数列: 95 38 70 88 16 42 45 68 33 24 94 41 40 10 29 40 87 28 7 75
偶数数列: 38 70 88 16 42 68 24 94 40 10 40 28
进程已结束，退出代码 0

```