JavaSE 基础

1：基础语法

1.1 数据类型

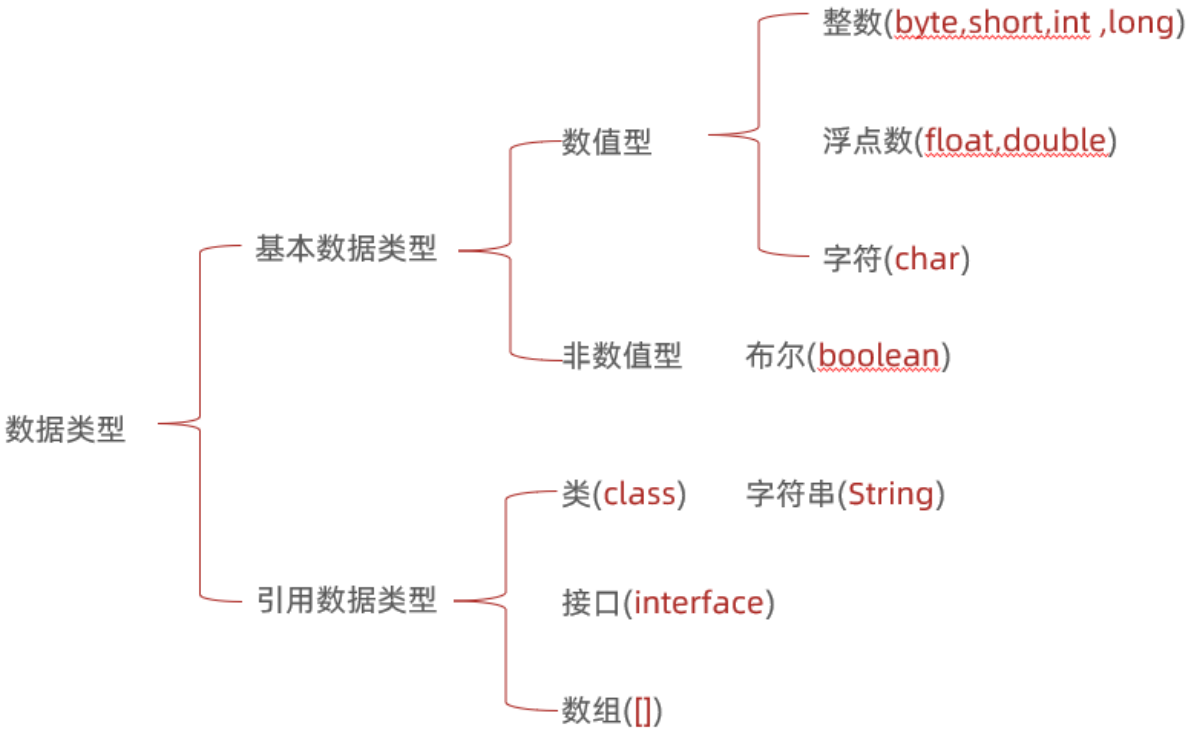
1TB = 1024GB

1GB = 1024MB

1MB = 1024KB

1KB = 1024B

这样呢，我们对字节就能有一个基础认知。有了一个基础的认知之后，我们再来说Java中的数据类型



在这里，我们给出每种基本数据类型的内存占用和取值范围，大家了解一下：

数据类型	关键字	占用内存	取值范围
整数	byte	1	-128~127
	short	2	-32768~32767
	int	4	-2的31次方到2的31次方-1
	long	8	-2的63次方到2的63次方-1
浮点数	float	4	负数：-3.402823E+38到-1.401298E-45 正数：1.401298E-45到3.402823E+38
	double	8	负数：-1.797693E+308到-4.9000000E-324 正数：4.9000000E-324到1.797693E+308
字符	char	2	0~65535
布尔	boolean	1	true，false

说明：E+38表示：乘以10的38次方。同理E-45表示：乘以10的负45次方

整数默认是：int类型

浮点数默认是：double类型

1.2 关键字

关键字的特点：

- 关键字的字母全部小写
- 常用的代码编辑器，针对关键字有特殊的颜色标记，非常直观

知道了关键字的特点后，这里我们给大家看看Java中的关键字：

abstract	assert	boolean	break	byte
case	catch	char	class	const
continue	default	do	double	else
enum	extends	final	finally	float
for	goto	if	implements	import
instanceof	int	interface	long	native
new	package	private	protected	public
return	strictfp	short	static	super
switch	synchronized	this	throw	throws
transient	try	void	volatile	while

1.3 标识符

- 不能以数字开头
- 不能是关键字
- 区分大小写

命名约定：

- 小驼峰命名法
  - 约定1：标识符一个单词的时候，首字母小写
  - 范例1：name
  - 约定2：标识符是多个单词的时候，第一个单词首字母小写，其他单词首字母大写
  - 范例2：firstName
- 大驼峰命名法
  - 约定1：标识符一个单词的时候，首字母大写
  - 范例1：Hello
  - 约定2：标识符是多个单词的时候，每个单词首字母大写
  - 范例2：HelloWorld

一般来说，小驼峰命名法适用于对方法，变量等起名字。大驼峰命名法适用于对类，接口等起名字。

最后再强调一点，就是我们起名字，最好能够做到：**见名知意**

2：运算符

注意事项

①/ 和 % 的区别：两个数据做除法，/ 取结果的商，% 取结果的余数

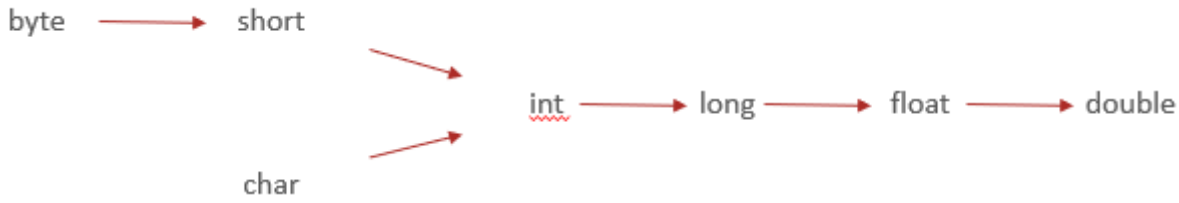
②整数操作只能得到整数，要想得到小数，必须有浮点数参与运算

2.1 数字相加(类型转换)

在Java程序中，数据参与运算，要求类型一致。这里就涉及到了数据的类型转换。而类型转换又分为两种：

1. 隐式转换
2. 强制转换

隐式转换：把一个表示数据范围小的数值或者变量赋值给另一个表示数据范围大的变量



强制转换：把一个表示数据范围大的数值或者变量赋值给另一个表示数据范围小的变量

格式：数据类型 变量名 = (目标数据类型)(数值或者变量)

2.2 字符相加

2.3 字符串相加

ASCII码表，里面有常用的字符及其对应的数值关系：

ASCII 码	键盘	ASCII 码	键盘	ASCII 码	键盘	ASCII 码	键盘
27	ESC	32	SPACE	33	!	34	"
35	#	36	$	37	%	38	&
39	'	40	(	41	)	42	*
43	+	44	'	45	-	46	.
47	/	**48**	**0**	49	1	50	2
51	3	52	4	53	5	54	6
55	7	56	8	57	9	58	:
59	;	60	<	61	=	62	>
63	?	64	@	**65**	**A**	66	B
67	C	68	D	69	E	70	F
71	G	72	H	73	I	74	J
75	K	76	L	77	M	78	N
79	O	80	P	81	Q	82	R
83	S	84	T	85	U	86	V
87	W	88	X	89	Y	90	Z
91	[	92	\	93	]	94	^
95	_	96	`	**97**	**a**	98	b
99	c	100	d	101	e	102	f
103	g	104	h	105	i	106	j
107	k	108	l	109	m	110	n
111	o	112	p	113	q	114	r
115	s	116	t	117	u	118	v
119	w	120	x	121	y	122	z
123	{	124	|	125	}	126	~

字符串相加

- 当“+”操作中出现字符串时，这个“+”是字符串连接符，而不是算术运算
  - “zhongguo”+ 666
- 当连续进行“+”操作时，从左到右逐个执行
  - 1 + 9999 + “岁zhongguo”

2.5 赋值运算符

Java中的赋值运算符有如下几种：

符号	作用	说明
=	赋值	a=10，将10赋值给变量a
+=	加后赋值	a+=b，将a+b的结果赋值给a
-=	减后赋值	a-=b，将a-b的结果赋值给a
*=	乘后赋值	a*=b，将a*b的结果赋值给a
/=	除后赋值	a/=b，将a/b的结果赋值给a
%=	取余后赋值	a%=b，将a%b的结果赋值给a

我们来总结一下，刚才在演示代码的时候有个注意事项：

注意事项

扩展的赋值运算符隐含了强制类型转换

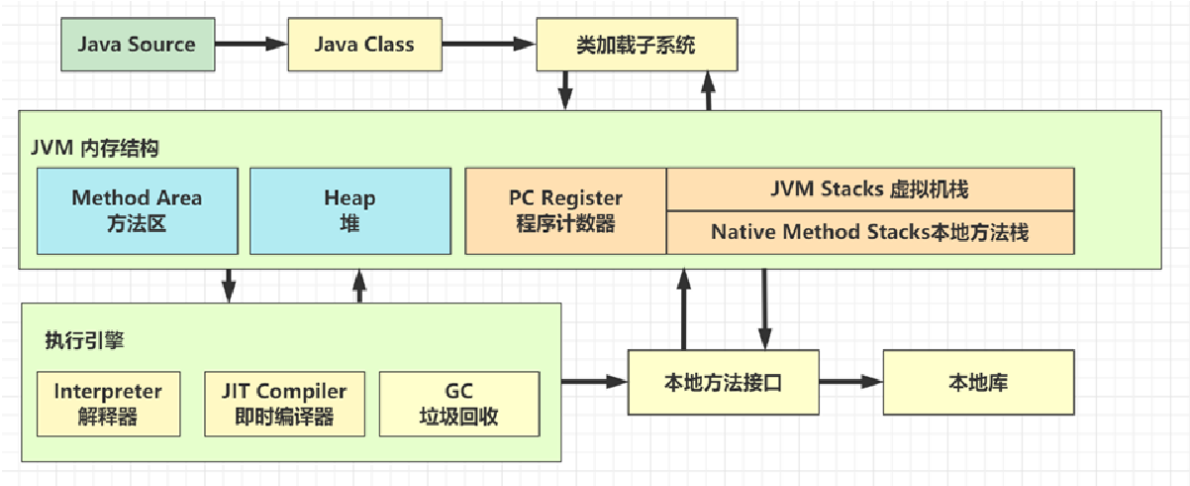
3: 方法

3.1 Debug查看方法调用

按照 Debug 的正常使用即可，但是要注意如下事项：

- 进入一个方法的时候，需要用 Step Into F7
- 在方法内部，看每一行代码的执行流程，使用 Step Over F8
- 注意观察方法的出现和消失，以及变量的变化

4: Java 内存分配



Java 内存分配

栈：所有局部变量都会在栈内存中创建

- 局部变量：定义在方法中的变量或者方法声明上的变量
- 方法执行都会加载到栈中进行
- 局部变量特点：随着方法的调用而存在，随着方法的调用完毕而消失

- 代码：`Student s = new Student();`

堆：所有对象及其对应的实例变量和数组都将存储在此处

- 简单理解为：new出来的东西，都存储在堆内存
- 每一个new出来的东西都有一个地址值，使用完毕，会在垃圾回收器空闲时被回收
- 实例变量(成员变量)有初始化值：
  - 基本数据类型：整数：0，浮点数：0.0，布尔：false，字符：空字符
  - 引用数据类型：null

5：成员变量和局部变量的区别

区别	成员变量	局部变量
类中位置不同	类中方法外	方法内或者方法声明上
内存中位置不同	堆内存	栈内存
生命周期不同	随着对象的存在而存在，随着对象的消失而消失	随着方法的调用而存在，随着方法的调用完毕而消失
初始化值不同	有默认的初始化值	没有默认的初始化值，必须先定义，赋值，才能使用

6：this关键字

this关键字

① this限定的变量用于指代成员变量

- 方法的形参如果与成员变量同名，
  不带this修饰的变量指的是形参，而不是成员变量
- 方法的形参没有与成员变量同名，
  不带this修饰的变量指的是成员变量

② 什么时候使用this呢？**解决局部变量隐藏成员变量**

③ this：方法被哪个对象调用，this就代表哪个对象

```
public class Student {
  private String name;

  public String getName() {
    return name;
  }
  public void setName(String name) {
    this.name = name;
  }
}
```

```
public class StudentDemo {
  public static void main(String[] args) {
    Student s1 = new Student();
    s1.setName("林青霞");

    Student s2 = new Student();
    s2.setName("张曼玉");
  }
}
```

7：构造方法的注意事项

构造方法的注意事项

① 构造方法的创建

- 如果没有定义构造方法，系统将给出一个**默认**的**无参数构造方法**
- 如果定义了构造方法，系统将不再提供默认的构造方法

② 构造方法的重载

- 如果自定义了带参构造方法，还要使用无参数构造方法，就必须再写一个无参数构造方法

③ 推荐的使用方式

- **永远提供无参数构造方法**