Java类中域和方法设置中的常见错误Java认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/629/2021_2022_Java_E7_B1_BB_E4_B8_AD_c104_629773.htm 在Java程序中，类是其实现功能的核心。如果说开发Java应用程序，就是一个类的构建与使用的过程这一点都不为过。可惜的是，不少程序员在编写类的时候，还是会犯一些常规的错误。笔者就对此做一个总结，望各位读者能够引起重视。 一、基本数据类型的初始化问题。 在Java语言中，跟其他开发语言一样，都定义了一些基本的数据类型。程序员可以拿来直接使用，还可以用来构造其他一些复杂的应用程序。不过在 Java中使用这些基本数据类型有些不同。在Java语言中，这个基本数据类型主要用在两个地方，分别为类中的成员或者局部变量。对于这些基本的数据类型，常犯的错误就是没有进行初始化。 如果将某些变量是属于这些基本数据类型的，当他们作为一个类的成员使用时，默认情况下编译器会给其一个默认值。以保证那些是基本类型的成员变量得到初始化，防止产生程序的错误。但是，这些默认的初始值大部分情况下可能都不符合程序员的需要，可能是不准确的，也可能是不合法的。为此笔者的建议是，开发人员要养成一个好习惯，在定义变量的时候(用于类成员)，就最好明确的对变量进行初始化。另外需要注意的是，跟C 等编程语言相比，Java在这方面有了很不错的改善。如如果开发人员忘记对其进行初始化，Java至少还会采用默认值对其进行初始化，来防止程序错误。但是在C 中，则不会对其进行自动初始化。在这种情况下，由于变量没有及时初始化，而很容易出现程序的崩溃。 基本类型的另外一种用途

就是当作局部变量来使用，如在循环语句中当作循环条件来使用。此时跟类成员不同，系统不会自动对局部变量进行初始化。很多程序开发员会在这里栽跟头。因为在类成员中会对其进行自动初始化，而在局部变量也如此。其实不然。当利用java认证更多详细资料int y 等语句定义了一个局部变量时，Java跟其他开发语言一样，是不会对其进行自动初始化的。开发人员需要在定义变量时就对其进行初始化，这是笔者强烈建议的。如果在代码编译的时候，局部变量没有被正确的初始化，则系统编译器就会返回一个错误信息，告诉开发人员某个局部变量没有被正确的初始化，便以不会成功。在这方面，Java跟其他语言也有所不同。如在C 中，如果变量没有初始化，那么系统只是抛出一个警告信息，便以仍然可以正常进行。这无疑后以后程序运行埋下了一颗定时炸弹。为此笔者认为，Java这种做法是比较安全的。当局部变量没有初始化时，系统会拒绝进行编译，而不是简单的只是一个警告信息。有时候，警告信息并不能够引起程序开发人员的重视。总之，基本数据类型无论是作为类成员变量来使用，还是作为局部变量来使用，程序开发人员都要养成一个好习惯，即在变量定义的时候马上进行初始化。即使不知道还赋予什么值合适，那么最好也是手工的赋予其一个默认值。等到变量使用时，在根据实际情况给其重新赋值。无论什么情况下，变量一定以就要对其进行初始化。 二、给方法设置合适的返回类型。 在Java类中，除了成员变量，最重要的就是方法了。而在方法中，比较容易出现问题的那就是其返回类型的问题。这里指的返回类型是指调用方法后返回的数据类型。也就是说，在方法内部执行一系列的运算之后，要返回给外部

的值。开发人员要根据这个值来设置方法的返回类型。如现在有一个方法 age()。主要的用途就是根据身份证号码来推算出这个人的年龄。此时在调用这个方法时，可利用如下的语句 int age1=getage.age()。 那么在定义这个方法的时候，需要注意哪些内容呢? 首先在定义方法的时候，需要注意其数据的范围类型。并不是说所有的方法都会返回结果。如果某个方法返回结果的话，并不是说不用指定返回的数据类型。仍然需要指定，只是利用关键字void，而不是采用int等基本数据类型。如假设上面这个age方法没有返回任何的数据类型，则要按照如下的方法来定义，即void age()。注意前面的关键字void表示这个方法没有返回值。如果有返回值的话，那么就要在方法开头指明返回值的类型。如age这个方法，如果其返回的是年龄整数型的数据，则就需要如下定义int age()。最好的情况是，返回的数据类型跟方法中指定的数据类型一致。，但是在实际工作中，没有定义的这么严格。一般来说，这个数据类型只要兼容机即可。如虽然返回的数据是整数，但是在方法处定义的可以是浮点数的数据类型。因为他们是兼容的。但是反过来则不行。如返回的值为浮点数，而在方法中定义的却是整数型的返回值。此时系统就会认为是错误的。因为在整数型的变量中无法存储浮点数的数据。 其次需要注意的是在调用这个方法时，需要将方法的返回值赋值给某个变量。为此，这个变量的数据类型必须同这个方法返回值的数据类型兼容。int age1=getage.age()，如在这个语句中，如果方法age返回的是整数型的数据，就说明是可行的。但假设这个方法返回的不是整数型的数据，而是浮点数的数据类型，那么就会发生数据类型不兼容的情况。所以在将某个方

法的返回值赋值给变量时，一定需要注意这个方法的返回值到底是什么。如果这个方法是程序员自己编写的，那么一般不会出现问题。自己编写的方法，应该清楚其会返回什么样的值。现在的问题是，如果这个方法不是程序员自己开发的呢?在一些大型的应用程序开发中，往往需要多个程序员合作才能够完成一个项目。此时程序员就需要引用其他程序开发人员开发的类以及类中的方法。有时候程序开发人员还会从网上下载类直接拿来使用。无论是哪一种方式，都会涉及到这个数据类型的问题。为此如果是程序员自己开发的类供别人使用时，那么最好能够在类或者方法的说明中，将这个方法会返回的数据类型写清楚，并且要确保准确。只有如此，程序员才能够在引用某个方法时为其设置合适的数据类型，避免因为数据类型不一致而导致应用程序编译或者运行错误。 三、参数列表的个数与数据类型。 在定义方法时，往往需要外部参入一些参数才能够运行。如在根据身份证号码来判断年龄的方法中，至少需要将身份证号码这个参数传递放方法对内部中去。有时候可能还不止一个参数。如由于身份证号码新版与旧版是不同的。为此同时需要将身份证号码的版本传递到方法里面去。在这个传递的过程中，主要需要注意的就是参数的个数与数据类型。 如果外部需要传入多个参数到方法内部的话，那么在定义方法时就需要设置这么多的参数，传入的参数与参数列表中的参数必须一一对应。如上面这个案例中需要传入身份证号码与身份证的版本，那么在定义这个方法的时候就需要设置两个参数。 其次需要注意数据类型。在参数列表中，也需要为参数指定其采用的数据类型。这个数据类型必须与实际传入的数据类型一致或者兼容。

为了保证这一点，往往需要在传入这个参数之间进行控制。如需要利用一个If语句来判断传入的参数是否符合要求。如果符合的话，调用这个方法。相反不符合的话，则会抛出一个异常，并将错误信息返回给用户。在大部分情况下，如果方法涉及到参数，利用if语句来进行判断是比较合理的。另外，也可以事先利用数据类型转换函数，强制将传输参数的数据类型转换成参数列表所定义的数据类型。不过需要注意的是，在转换过程中可能会造成一些数据的误差。开发人员需要预先评估一下能否接受这误差。如果可以接收的话，则可以采用这种方法来强制数据的兼容性。如果不行的话，则就需要给用户返回错误信息，让其提供一个合法的参数。 另外在窗口中进行类似的控制也是不错的选择。如可以通过必须填写文本框这种机制来控制用户必须传入多少个参数。或者在文本框中采用列表或者日期型的选择框等等来规范用户传入参数的数据类型。总之，这个方法传入参数的控制，无论是数量上还是数据类型上，在越靠近用户的那边控制效果越好。如此用户可以在最短的时间内发现问题，而不会等到程序执行错误了才知道自己提供的参数有问题。 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com