

术语：

新 运营商类对象

点 (。) 运算符参数返回
别名垃圾收集String类不可
变的Java API包

进口报关Random类Math类N
umberFormat类DecimalForm
at的类包装类自动装箱拆箱

参考变量实例零位法静态
方法构造

课程编码标准加入

- 当使用进口报关，从单独同一个包导入类（即做 不 使用*符号）。例如，你应该有 进口java.util.Scanner中 和 导入了java.util.Random 而不是 进口的java.util.* 其中进口一切都在 java.util中 包。
- 如果是在 π 的计算，使用Math.PI而不是字面值，如3.141。你应该总是使用常量可用时。
- 从现在开始，使用扫描仪对象何时扫描输入（例如，从键盘）时，使用nextLine从用户获得数字输入并将其转换成使用从相应的包装类方法的一个数字。例如，要在一个读 INT 从扫描仪userInput值：

```
INT敏=的Integer.parseInt ( userInput.nextLine ( ) );  
例如在读 双 从扫描仪扫描值：  
双myDouble = Double.parseDouble ( userInput.nextLine ( ) );
```

目标：

通过本次活动结束后，你应该能够做到以下几点：

- Ø 了解如何使用实例化对象 新 操作者
- Ø 使用Java标准库 (在Java API) 来查找类
- Ø 使用String类和扫描仪类
- Ø 构建一个观众画布在画布上运行您的程序

项目描述：

您将创建一个交流，包含字母对消息进行编码的程序。

在实验室路线：

当你逐步建立你的程序通过在每个下面的代码段，你应该编译和各段完成后运行程序。这将允许你来验证你的程序每一个步骤之后是正确的，因为你朝着完成计划的进展情况。

MessageConverter.java

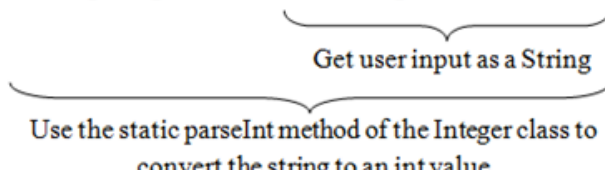
- 这项计划将在来自用户的信息读出，然后要求用户输入一个数字。会发生以下情况根据用户输入的数字：
 - Ø 如果用户输入一个1，该消息将被打印修剪。
 - Ø 如果用户输入一个如图2所示，该消息将在小写打印。
 - Ø 如果用户输入一个3，该消息将在上壳体被打印。
 - Ø 如果用户输入一个如图4所示，该消息将被打印与替换所有的元音下划线。
 - Ø 如果用户输入一个如图5所示，将被打印的消息，而无需在第一和最后一个字符。
 - Ø 任何其它数目应产生一个适当的消息。
- 创建一个名为类 的**MessageConverter** 包括主方法。另外添加一个import语句java.util.Scanner的类。import语句应该是在Java文件的第一行 (即，说到类的声明以及评论之前) 。
- 在main方法，声明下面的变量：
 - userInput：用于从键盘读取用户输入扫描仪对象。
扫描仪userInput = 新 扫描仪 (System.in);
 - 信息：用于用户输入的字符串对象; 初始化为空字符串。
字符串消息= "";
 - 输出类型：表示用户想要的输出类型的int。
 - 结果：为转换的消息字符串对象; 初始化为空字符串。
字符串结果= "";
- 从用户请求输入，则使用扫描仪类的nextLine () 方法获得用户输入。

```
System.out.print("Type in a message and press enter:\n\t> ");  
message = userInput.nextLine();
```

- 现在，得到了用户的输出类型的输入。请确保您了解下面的代码工作。请注意，这是打印方法，而不是一个的println调用。

```
System.out.print("\nOutput types:"
    + "\n\t0: As is "
    + "\n\t1: Trimmed"
    + "\n\t2: lower case"
    + "\n\t3: UPPER CASE"
    + "\n\t4: v_w_ls r_pl_c_d"
    + "\n\t5: Without first and last character"
    + "\nEnter your choice: ");

outputType = Integer.parseInt(userInput.nextLine());
```



- 设置一个 *如果* 声明将打印出基于用户选择的输出类型相应的消息。请注意，您可以在替代条件下编写代码 *如果* using 语句

否则，*如果* 之后 *如果* 块。

```
if (outputType == 0) { // as is
}
else if (outputType == 1) { // trimmed
}
else if (outputType == 2) { // lower case
}
else if (outputType == 3) { // upper case
}
else if (outputType == 4) { // replace vowels
}
else if (outputType == 5) { // without first and last character
}
else { // invalid input
}
```

- 后 该 整个 如果 别的 声明中，添加以下代码行打印出来的结果。

```
System.out.println("\n" + result);
```

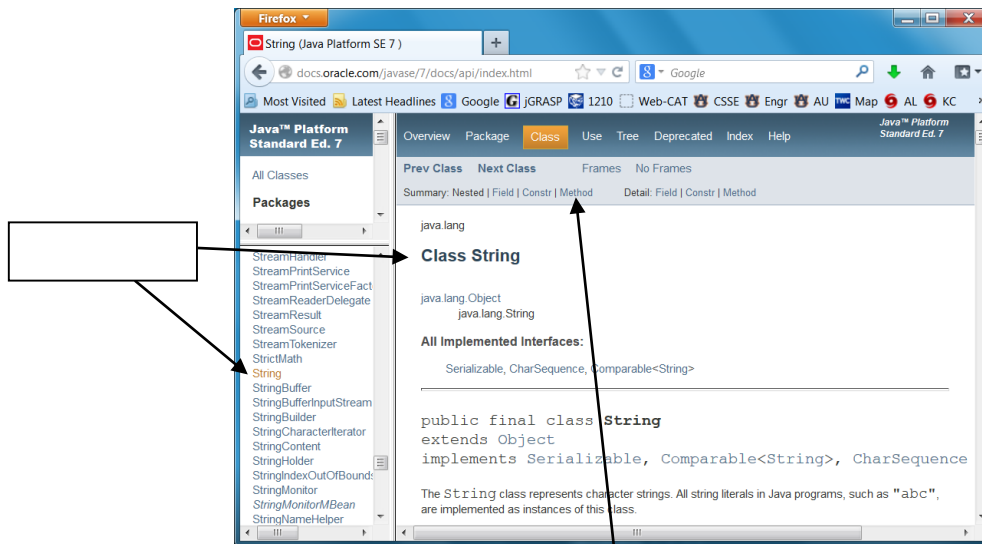
现在编译和下面的每个步骤之后 如您在填写 *如果*-否则，如果其他在下面的步骤。

- 为输出类型等于0的第一条件设定结果到原始消息，其可包括前缘和/或后的空格。

```
if (outputType == 0) { // as is
    result = message;
}
```

- 由于消息引用String类的一个实例，我们可以使用String类的方法返回一个新的String是通过消息中引用原字符串的修改版本（原来的String对象没有改变）。要查看String类可用的方法，转到Java API在 <http://docs.oracle.com/javase/8/docs/api/> 并找到

串 类网页并在其上用鼠标左键点击左下方的窗格中。这将在API的大型右边的窗口中打开页面的字符串，如下图所示。因为它描述了所有附带的JDK类的，你应该熟悉API。



以是原始字符串的修改版本。String类

- 在右边主窗口中，单击 **方法** 在顶部或向下滚动到 **方法摘要** 看到这些方法，我们的String对象 信息 可以使用。
- String方法不修改它们被称为String对象。相反，如果一个方法有一个字符串返回类型，则返回 一个新的String对象 其可

的字符串方法列表包括
返回类型

和 描述
的每个方法。例如，

的charAt 方法返回一个
烧焦
在指定的值 指数。

- 为输出类型等于1次将结果提供给消息的修整值的第二条件; 也就是说, 我们可以用修剪方法删除任何开头或结尾的空白。一个方法的API条目显示在左侧的方法与形式参数, 如果有的话, 就用的方法所做的简要说明沿着正确的方法名返回类型。因此, 如下所示的条目 **修剪** 方法表示它返回一个字符串并且没有参数。记得, 该修剪方法不修改在其它被称为字符串对象。相反, 它会返回一个新的String对象, 它是原始的字符串的修剪版本。

返回字符串

方法名称 - 有这个方法没有任何形式参数

String

trim()

Returns a string whose value is this string, with any leading and trailing whitespace removed.

一定要编译 每次完成的一部分 **如果-否则, 如果其他** 下面。

```
else if (outputType == 1) { // trimmed
    result = message.trim();
}
```

- 第三个条件用于输出类型等于2次将结果提供给消息的下壳体值; 也就是说, 我们可以用与toLowerCase方法。对于下面所示的Java API入口

toLowerCase 方法表示它返回一个字符串并且没有参数。

返回字符串

方法名称 - 有这个方法没有任何形式参数

String

toLowerCase()

Converts all of the characters in this String to lower case using the rules of the default locale.

基于这一认识, 你现在可以调用与toLowerCase方法对我们的 信息

目的是在小写返回其值。放置在if语句的第二块下面的代码:

```
else if (outputType == 2) { // lower case
    result = message.toLowerCase();
}
```

- 看看对结果与toUpperCase方法的API描述, 并把下面一行if语句的第二块:

```
else if (outputType == 3) { // upper case
    result = message.toUpperCase();
}
```

- 现在写的代码, 将与其他字符的消息中替换指定的字符。为了改变一个角色, 我们需要使用 **更换** 字符串的方法来改变我们的字符串的某些字符。看看的 **更换** 方法的Java的API中。我们看到, 它返回一个字符串, 它需要两个参数: 要替换的字符和新的字符。注意, 有取代的两个版本: 第一个采用类型的参数 **烧焦**

(例如, “一”或“E”)。与第二采用类型的参数为 `CharSequence` 其包括字符串文字 (例如, “一”或“ABC”)。在下面的代码中, 我们将使用 *烧焦* 版。

返回字符串

方法名称和两个形式参数

String	replace(char oldChar, char newChar) Returns a string resulting from replacing all occurrences of oldChar in this string with newChar.
String	replace(CharSequence target, CharSequence replacement) Replaces each substring of this string that matches the literal target sequence with the specified literal replacement sequence.

如同其他字符串的方法中, 更换方法不修改字符串对象它。相反, 它返回它的新特性已经取代旧符的新String对象。

- 首先, 使用 更换 方法所有A的更改为下划线。下面的代码行替换消息值的“a”字符, 然后分配消息的新版本引起的。再次注意, 消息本身保持不变。

```
else if (outputType == 4) { // replace vowels
    result = message.replace('a', '_');
}
```

现在, 通过将剩余的元音下划线的输出类型等于4完成翻译。

```
else if (outputType == 4) { // replace vowels
    result = message.replace('a', '_');
    result = result.replace('e', '_');
    result = result.replace('i', '_');
    result = result.replace('o', '_');
    result = result.replace('u', '_');
}
```

- 现在使用 子 方法和长度的方法来提取所述消息的一部分 (或子串)。返回字符串

方法名称和形式参数

String	substring(int beginIndex) Returns a string that is a substring of this string.
String	substring(int beginIndex, int endIndex) Returns a string that is a substring of this string.

回想一下在字符串中的字符为0被索引开始因为这里的要求是结果中包括除第一个和最后一个字符, 你应该使用的第二个版本

子用1作为参数beginIndex和message.length () - 1作为endIndex的。

```
else if (outputType == 5) { // without first and last character
    result = message.substring(1, message.length() - 1);
}
```

- 最后，如果用户输入大于1时，其他的数2, 3, 4, 或5, 结果应该被设置为一个错误信息。

```
else { // invalid input}
    result = "Error: Invalid choice input.";
}
```

- 根据测试每个条件程序。

原版的（“原样”）：对于该消息，输入：“这是一个测试。”
一定要包括五个前导空格，但省略了引号。对于选择，输入0。

在一个信息，然后按类型输入：

> 这是一个测试。

输出类型：

0：由于是1：修剪2：下壳体3：大写

4：v_w_ls_r_pl_c_d

5：如果没有第一个和最后一个字符输入您的选择：0

这是一个测试。

修剪：使用向上箭头键来获取你在上面输入的信息，然后选择输入1.注意五个前导空格在输出被删除。

在一个信息，然后按类型输入：

> 这是一个测试。

输出类型：

0：由于是1：修剪2：下壳体3：大写

4：v_w_ls_r_pl_c_d

5：如果没有第一个和最后一个字符输入您的选择：1

这是一个测试。

小写： 使用向上箭头键来获取你在上面输入的信息，然后选择进入2.注意输出为小写，但五大主导空间有 不 被删除。

在一个信息，然后按类型输入：

> 这是一个测试。

输出类型：

0：由于是1：修剪2：下壳体3：大写

4：v_w_ls_r_pl_c_d

5：如果没有第一个和最后一个字符输入您的选择： 2

这是一个测试。

大写： 使用向上箭头键来获取你在上面输入的信息，然后选择，输入3.注意输出为全大写，但五大主导空间有 不 被删除。

在一个信息，然后按类型输入：

> 这是一个测试。

输出类型：

0：由于是1：修剪2：下壳体3：大写

4：v_w_ls_r_pl_c_d

5：如果没有第一个和最后一个字符输入您的选择： 3

这是一个测试。

元音取代： 使用向上箭头键来检索您在上面输入，然后选择邮件，输入4.注意输出具有元音删除，但五大主导空间有 不 被删除。

在一个信息，然后按类型输入：

> 这是一个测试。

输出类型：

0：由于是1：修剪2：下壳体3：大写

4：v_w_ls_r_pl_c_d

5：如果没有第一个和最后一个字符输入您的选择： 4

TH_S _s _t_st。

如果没有第一个和最后一个字符：对于该消息，进入“这是一个测试。” 这个时候不要输入任何开头或结尾空格。然后供选择，输入5.注意输出在第一和最后一个字符删除。

在一个信息，然后按类型输入：

> 这是一个测试。

输出类型：

0：由于是1：修剪2：下壳体3：大写

4：v_w_ls r_pl_c_d

5：如果没有第一个和最后一个字符输入您的选择： 五

他是一个测试

输入无效：

在一个信息，然后按类型输入：

> 这是一个测试。

输出类型：

0：由于是1：修剪2：下壳体3：大写

4：v_w_ls r_pl_c_d

5：如果没有第一个和最后一个字符输入您的选择： 7

错误：无效的选择输入。

- 尝试输入消息“的消息”，然后选择用下划线代替所有的元音。你注意到什么关于输出？为什么你认为元音的一个没有被取代？你会如何去纠正这个问题？你不必做出这种改变 - 只是想想而已。

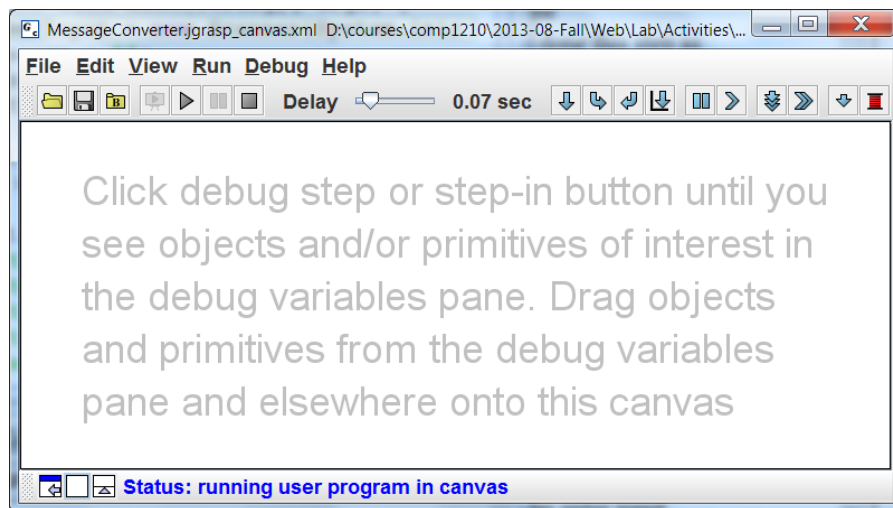
网络-CAT - 你们在这方面的活动，完成计划后，您应该提交您的



MessageConverter.java 文件对Web-CAT。这是你将提交本次活动的唯一文件。


使用浏览器画布与您的MessageConverter程序

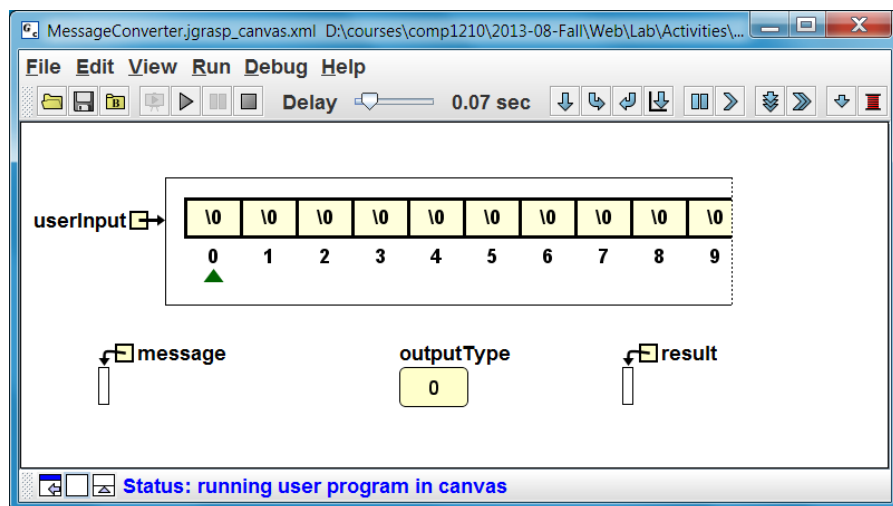
现在，我们要创建一个观众的画布，所以我们可以看到在MessageConverter的程序一步一步发生的事情，因为它运行。请注意，我们可以尽快变量上面已经宣布完成这一步。为了您的项目任务，你应该考虑在你需要深入了解到底是什么在你的程序中发生的任何点创建的画布。


- 随着MessageConverter.java在CSD编辑窗口中，单击工具栏上的按钮，画布的运行。在CSD窗口中，你应该看到该程序停止在第一个可执行语句，你应该看到一个空的浏览器窗口的画布，如下图所示。



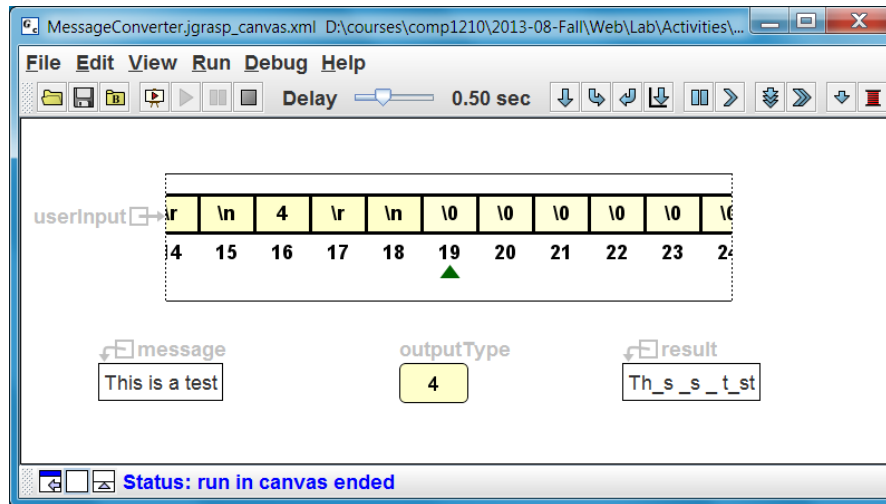
- 如通过在画布的消息中所指示，点击调试步骤按钮  执行语句来创建System.in一个扫描对象，它是键盘。您现在应该看到变量名 userInput 在调试选项卡。现在，单击步骤按钮  更多次，让你看到 消息，因此，和 输出类型 在调试选项卡。现在拖动这些变量到画布窗口，并安排他们见下图。单击保存按钮

 画布工具栏上。



- 现在点击播放按钮 ，你的程序将开始自动步进，直至暂停等待输入。你每次进入要求输入后，在画布上的观众将被更新。该 运行帆布 将持续到你的程序终止其时如在下面的例子在画布上的观众展示他们的最后的值

信息“这是一个测试”和 输出类型=4。



虽然我们创造了画布上的MessageConverter程序完成后，它会一直更适合作为方案正在编码，创建画布，特别是如果你在程序中遇到的逻辑错误。例如，如果你感觉到，在程序特定地方发生了错误，你应该要检查行为的语句设置一个断点。[要设置断点，将鼠标移动到边缘到语句的左边，直到你看到红色的八角形（停止标志符号），然后左键单击鼠标。然后，而不是点击画布上运行按钮的 在工具栏上，单击调试按钮



断点并停止。你可以单步



（或者步骤-in）和观察变量

在画布和调试标签，以帮助确定错误的性质，这样就可以纠正。



该程序将运行到