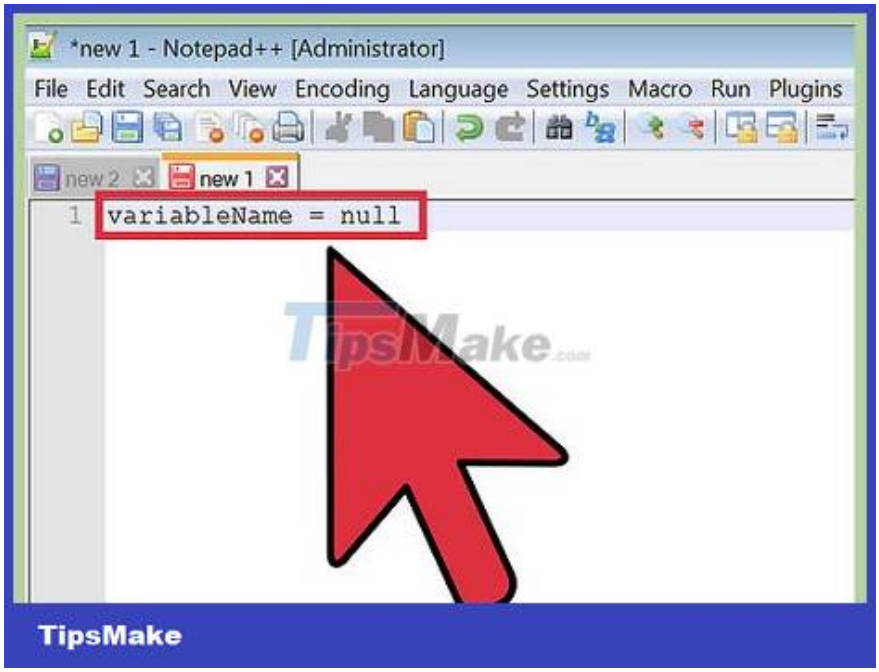
How to Check for Null in Java

Null represents variables that do not refer to any object and contain no value. You can use a basic 'if' statement to test for null values ??in a piece of code. Null is often used to represent or determine the nonexistence of some value. In such a context, null can be used as a condition to start or stop other processes in the code.[1] X Research source

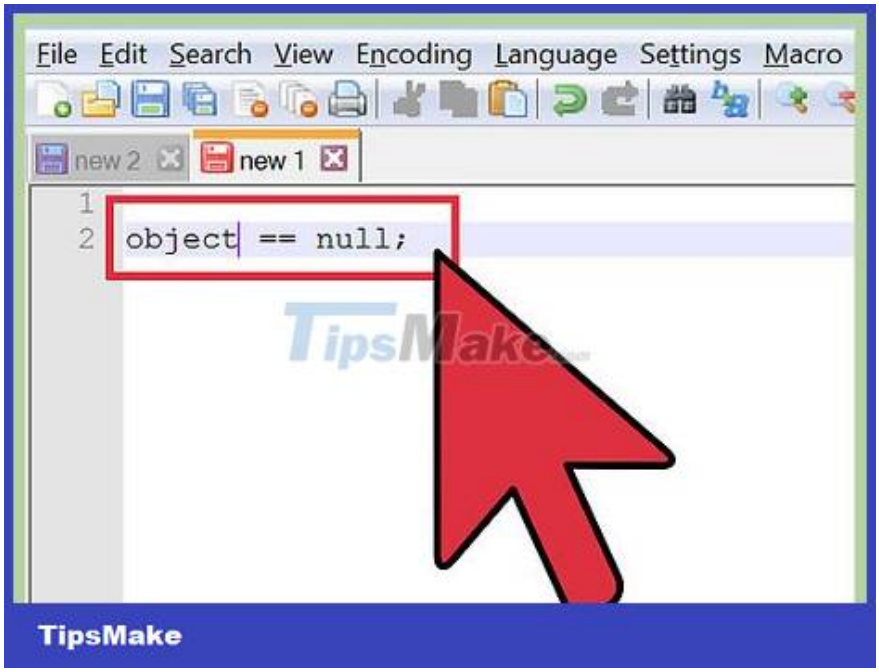
Checking for null in Java



Use the '=' sign to define a variable. An '=' sign is often used to declare and assign values ??to variables. You can use this to set a variable to null.

The values ??of '0' and null are not the same, so their uses are also different.

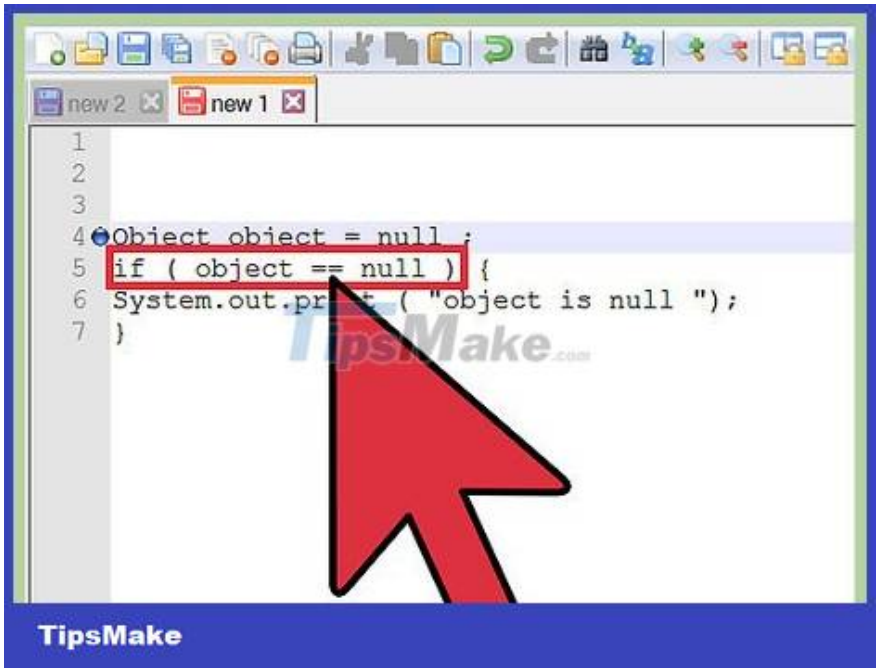
```
variableName = null;
```



Use two '==' signs to check the value of the variable. The two '==' signs are used to check whether two values ??on different sides are equal. If you assign a variable null with an '=' sign, then when you check whether the variable is null or not, you will get the value true.

```
variableName == null;
```

You can also use the '!=' operator to test if some value is NOT equal.

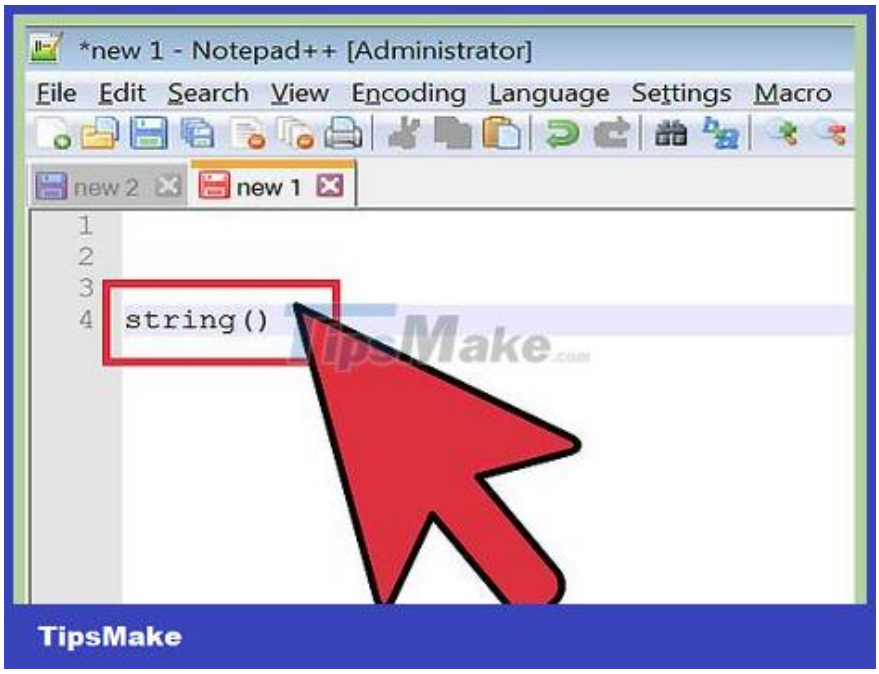


Use 'if' statement to condition null variable. The result of the expression will be a boolean value (true or false). You can use a boolean value as a condition for the statement to execute afterward.

For example, if the value is null, you enter 'object is null'. If the two '==' signs do not find that the variable is null, the operator will ignore the condition or find another path.

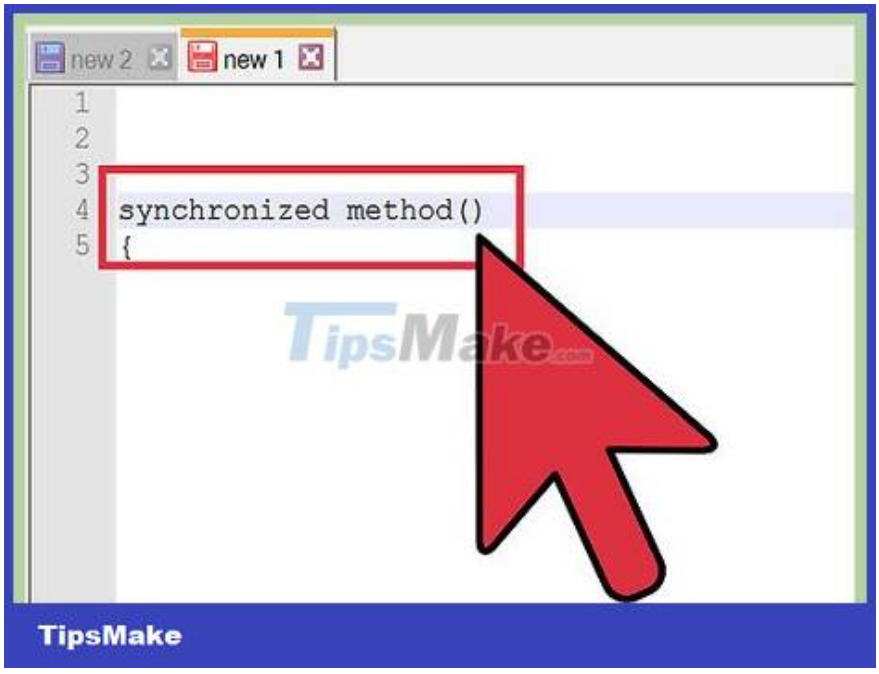
```
Object object = null ; if ( object == null ) { System.out.print( "object is null
```

Use the null check statement

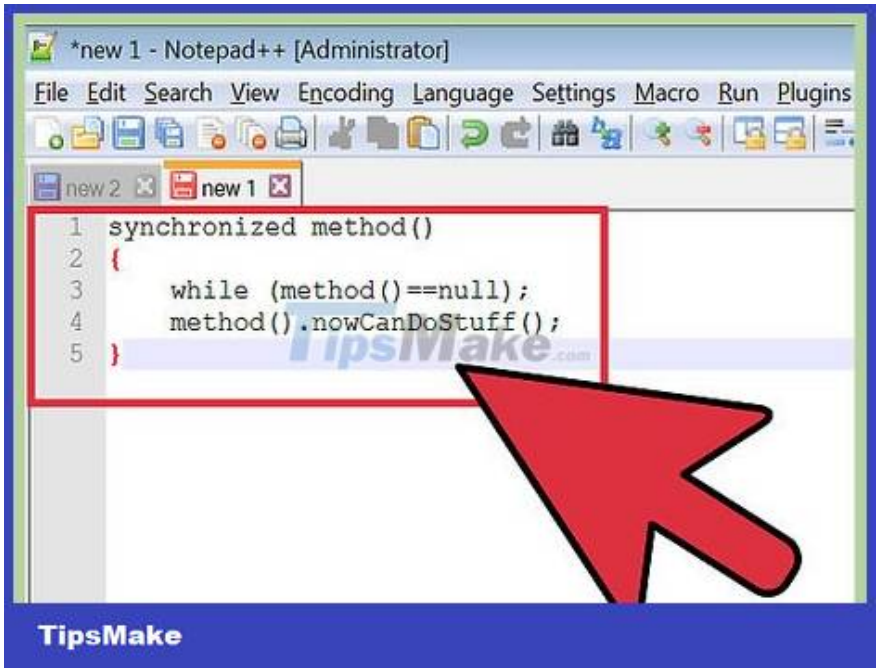


Use null as unknown value. Null is very often used as a default setting in place of any assigned value.

With `string()`null will be a temporary value until that value is actually used.



Use null as a condition to end the process. Returning null can be used to create loop termination or pause the process. This is often applied to throw errors or exceptions when something goes wrong or when an unexpected condition occurs.



Use null to indicate uninitialized state. Similarly, null can be used as an indication that the process has not yet started or as a condition to mark the start of the process.

For example, execute something while the object is null, or do nothing until the object is NO longer null.

```
synchronized method() { while (method()==null); method().nowCanDoStuff(); }
```

You finished reading the article "**How to Check for Null in Java**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.
