# Runtime type detection

Question
Sean Connelly · Aug 25, 2017

# Runtime type detection

SOLVED

## tl;dr how can you tell if a number is really a string

*The original question has been updated/improved.*

Equality comparisons on floating point numbers will produce different results...

"1.1"=1.1  //is true!

"0.1"=0.1  //is not true :(

This second comparison can be fixed with...

+"0.1"=+0.1 // is true!

The problem is, what if we don't realise that a value is a stringy number, or just overlook implementing this defensive check.

One solution would be to lint check %Float properties and return types that should originate from a number and not a string, as discussed here...

https://community.intersystems.com/post/compilation-gotchas-and-request-change

A second approach is to use Unit Testing to ensure not only the values of a test are the same, but also the types of those values.

If for instance, a method should return a value of type %Float, then instead of using a normal AssertEquals() method, the unit test could implement an AssertFloatEquals() or AssertNumberEquals() which would check the return value is a pure number and not a stringy number. This would fix problems upstream before they can happen.

So, boiling all of this down, how can you tell if a number is really a string.

A simple condition for the solution should produce a false (zero) for both of these tests

$$$AssertNumberEquals("0.1",0.1)
$$$AssertNumberEquals("1.1",1.1)

**Answers that are not hitting the mark...**

1. Implement (+a=+b) in the AssertNumberEquals() method.

This will create a false positive test for (+"0.1"=+0.1), the point is that these need to fail. It also opens up tests to incorrectly pass values such as "1B" and 1.

2. Use "sort by", such that "0.12345"]]$c(0) returns 1 and 0.12345]]$c(0) returns zero.

Whilst a brilliant and innovative answer, it turns out that it of course only works for floating numbers with a leading zero.

It also turns out that $length(+num)=$length(num) will also do the same thing without the collation problems described below.

3. Use $IsValidNum

Whilst this will determine if a string contains a valid number, it does not tell us if the number is contained within a string.

4. Use ["0.12345"].%GetTypeOf(0) which will return "string"

I got this to work with the latest versions of Caché, but I was unable to find anything that was backwards compatible.

#Object Data Model #ObjectScript #Caché
00 2 0 31 889

Log in or sign up to continue
Add reply

**Source URL:** https://community.intersystems.com/post/runtime-type-detection