Article
Eduard Lebedyuk · Aug 9, 2016 2m read

Tips & Tricks - Process-private Globals as a class storage

Process-private Globals can be used as a data global in storage definition. That way, each process can have its own objects for the class with ppg storage. For example lets define a pool, which can:

- add elements to a pool (ignoring duplicates)
- check if an element exists in the pool

Here's the class:

```
/// Stores unique identifiers
Class Utils.Pool Extends %Persistent
{

Property Value As %String;

Index IDKEY On Value [ IdKey, PrimaryKey, Unique ];

Method %OnNew(Value As %String = "") As %Status [ Private, ServerOnly = 1 ]
{
    Set ..Value = Value
    Quit $$$OK
}

ClassMethod Add(Value As %String = "")
{
    Quit:..%ExistsId(Value)
    Set Obj = ..%New(Value)
    Do Obj.%Save()
}

Storage Default
{
<Data name="PoolDefaultData">
<Value name="1">
<Value>%%CLASSNAME</Value>
</Value>
</Data>
<DataLocation>^||Utils.PoolD</DataLocation>
<DefaultData>PoolDefaultData</DefaultData>
<IdLocation>^||Utils.PoolD</IdLocation>
<IndexLocation>^||Utils.PoolI</IndexLocation>
<StreamLocation>^||Utils.PoolS</StreamLocation>
<Type>%Library.CacheStorage</Type>
}
}
```

And the testing code:

```
/// Do ##class(Utils.Pool).Test()
ClassMethod Test()
{
    Do ..%KillExtent()
    Do ..Add(1)
    Write "Is 1 in pool: ", ..%ExistsId(1),!
    Write "Is 2 in pool: ", ..%ExistsId(2),!
    Do ..Add(2)
    Write "Is 2 in pool: ", ..%ExistsId(2),!
}
```

Outputs:

```
Is 1 in pool: 1
Is 2 in pool: 0
Is 2 in pool: 1
```

This approach can be used if you want:

- Storage space, inaccessible to other processes (without using locks, etc)
- Store something only while process lives

The code is also available on GitHub.

#Code Snippet #Data Model #Globals #Object Data Model #ObjectScript #Tips & Tricks #Caché

Source URL: https://community.intersystems.com/post/tips-tricks-process-private-globals-class-storage