***Caché***

***and***

***MUMPS***

***Part II***

***Examples and Tables***

This document is provided as an aid in downloading examples. It is intended as an addendum to the book *Caché and MUMPS – Part II* and contains no explanations of the examples in and of itself.

Example Code Line

|

| Code line

|

Example Variable Example

|

| Kill

| Set ^PERSON = "DAVID – DATA ABOUT DAVID"

| Set ^PERSON2 = "MICHAEL – DATA ABOUT MICHAEL"

|

Example Array Example

|

| Kill

| Set ^PEOPLE("DAVID")="DATA ABOUT DAVID"

| Set ^PEOPLE("SUSAN")="DATA ABOUT SUSAN"

| Set ^PEOPLE("MICHAEL")="DATA ABOUT MICHAEL"

| Set ^PEOPLE("AMY")="DATA ABOUT AMY"

|

Table Differences between Global Arrays, Global Variables, Local Arrays, and Local Variables

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | One  Dimensional  (No Subscripts) | Multi–  Dimensional  (Subscripts) | Persistent,  Permanent  on Disk | Temporary,  reside only in memory |
| Global Arrays |  | Yes | Yes |  |
| Global Variables | Yes |  | Yes |  |
| Local Arrays |  | Yes |  | Yes |
| Local Variables | Yes |  |  | Yes |

Example Setting a Global Array

|

| ; it may help to look through the

| ; **Error! Reference source not found.** on the Kill command

| Kill ; kill all variables

| Kill ^Pets ; kill the ^Pets Global

| Set ^Pets="My Pet Company"

| Set ^Pets("Dogs")="Jack's Dog Service"

| Set ^Pets("Dogs","Rover")=""

| Set ^Pets("Dogs","Bud")=""

| Set ^Pets("Cats")="Jill's Cat Service"

| Set ^Pets("Birds")="Feather Inc."

| Set ^Pets("Fish")="Underwater Demolition"

| Set ^Pets("Turtles")="Shells Anonymous"

| Set ^Pets("Turtles","HardShell")=""

| Set ^Pets("Turtles","Mr. Softy")=""

| ZW ^Pets ;ZWrite command (see *Appendix E*)

|

Example Accessing a Global array with the $O command

|

|ProcessPets1 ; Routine Name

|  
| Set S1="" Do { ;1st loop start

|   Set S1=$O(^Pets(S1)) ;get next S1

| If S1="" Quit ;no more S1s  
| Write !,"S1:",S1," – ",^Pets(S1)

| Set S2="" Do { ;2nd loop start

|   Set S2=$O(^Pets(S1,S2)) ;get next S2

| If S2="" Quit ;no more S2s

| Write !," S2:",S2," – ",^Pets(S1,S2)  
| } While S2'="" ;check if no more S2s  
| } While S1'="" ;check if no more S1s  
| Quit

|

Example ProcessPets2 with Forever Do method

|

|ProcessPets2 ; Routine Name

|

| Set S1=""

| For Do Quit:S1="" ;2 spaces after the For&Do

| . Set S1=$O(^Pets(S1))

| . If S1="" Quit

| . Write !,"S1:",S1," – ",^Pets(S1)

| . Set S2=""

| . For Do Quit:S2="" ;2 spaces after the For&Do

| .. Set S2=$O(^Pets(S1,S2))

| .. If S2="" Quit

| .. Write !," S2:",S2," – ",^Pets(S1,S2)

|

Example Output from ProcessPets 1 and 2 Routines

|

| Kill

| Do ^ProcessPets1 or ProcessPets2

S1:Birds – Feather Inc.

S1:Cats – Jill's Cat Service

S1:Dogs – Jack's Dog Service

S2:Bud –

S2:Rover –

S1:Fish – Underwater Demolition

S1:Turtles – Shells Anonymous

S2:HardShell –

S2:Mr. Softy –

Example Data Global of a Caché Database

|

| Kill

| .

| .

| .

| Set ^DB(101)=$LB("Name1","Address1","DOB1",... )

| Set ^DB(102)=$LB("Name2","Address2","DOB2",... )

| Set ^DB(103)=$LB("Name3","Address3","DOB3",... )

| Set ^DB(104)=$LB("Name4","Address4","DOB4",... )

| .

| .

| .

Example 9 Example of Indexed Globals

|

| Kill

| .

| .

| .

| Set ^DB("NameIndex","Name1")=101

| Set ^DB(("NameIndex","Name2")=102

| Set ^DB(("NameIndex","Name3")=103

| Set ^DB(("NameIndex","Name4")=104

| .

| .

| .

| Set ^DB("DOBIndex","DOB1")=101

| Set ^DB(("DOBIndex","DOB2")=102

| Set ^DB(("DOBIndex","DOB3")=103

| Set ^DB(("DOBIndex","DOB4")=104

| .

| .

| .

|

| Set ^DB("AddressIndex","Address1")=101

| Set ^DB(("AddressIndex","Address2")=102

| Set ^DB(("AddressIndex","Address3")=103

| Set ^DB(("AddressIndex","Address4")=104

| .

| .

Example Code to build an index for Widget

|

| Set S1="" Do {  
| Set S1=$O(^DB(S1))

| If '+S1 Quit  ;if S1 not numeric  
| Set Widget=$Li(^DB(S1),4)  
| Set ^DB("WidgetIndex",Widget)=S1  
| } While S1'=""

|

Example Syntax of a Process–Private Global

|

| ^||Global

|

Example Process–Private Globals

|

| ;initially no need to kill Process–Private Global

| Set ^||Global="Abc"

| Set ^||Global("SUB1")="Def"

|

| Write ^||Global

Abc

| Write ^||Global("SUB1")

Def

| ;no need to kill the Process–Private Global

| ;at the end of processing

| Quit

|

Example CacheTempUser Globals

|

| Kill

| Kill ^CacheTempUser.MyData

|

| Set ^CacheTempUser.MyData="Abc"

| Set ^CacheTempUser.MyData("SUB1")="Def"

|

| Write ^CacheTempUser.MyData

Abc

| Write ^CacheTempUser.MyData("SUB1")

Def

| Kill ^CacheTempUser.MyData

|

Example ^TMP($J) or ^TEMP($J)

|

| Kill ;kill any left over variables

| Kill ^TMP($J) ;clean up your allotment

| ;of ^TMP

|

| Set ^TMP($J)="Abc"

| Set ^TMP($J,"Sub1")="Def"

|

| Write ^TMP($J)

Abc

| Write ^TMP($J,"Sub1")

Def

|

| Kill ^TMP($J) ;clean up before you exit

| Quit

|

Example Merge command

|

| Kill

| Merge TO=FROM

|

| Merge ^Global1 = ^Global2

|

| Merge ^Global = Array

|

| Merge ^Global1(Sub1) = ^Global2

|

| Merge ^Global1(Sub1) = ^Global2(Sub1,Sub2)

|

Example Merge Command

|

| Kill

| Kill ^Pets1,Pets2

| Set ^Pets1("Dogs")="Jack's Dog Service"

| Set ^Pets1("Dogs","Rover")=""

| Set ^Pets1("Dogs","Bud")=""

| Set ^Pets1("Cats")="Jill's Cat Service"

|

| Set ^Pets2("Dogs","Rover")="Owned by Jack"

| Set ^Pets2("Dogs","Buddy")=""

| Set ^Pets2("Dogs","My Pal")=""

|

| Merge ^Pets1 = Pets2

| ZWrite ^Pets1 ;ZWrite command (see Appendix E)

|

| ^Pets1("Cats")="Jill's Cat Service"

| ^Pets1("Dogs")="Jack's Dog Service"

| ^Pets1("Dogs","Bud")=""

| ^Pets1("Dogs","Buddy")=""

| ^Pets1("Dogs","My Pal")=""

| ^Pets1("Dogs","Rover")="Owned by Jack"

|

Example Lock command

|

| Lock ^X

|

| Lock ^X(Sub1)

|

Example Multiple Lock commands

|

| Lock ^X,^Y,^Z(Sub)

|

Example Incremental Lock command

|

| Lock +^Y

|

Example Incremental Unlock command

|

| Lock –^Y

|

Example Unlock All command

|

| Lock

**|**

Table – Building Blocks of Object Technology

|  |  |
| --- | --- |
| Building Blocks of Object Technology | |
| (1) | Objects |
| (2) | Methods |
| (3) | Class |

Example MyClass.Person

|

|Class MyClass.Person Extends (%Persistent,%Populate,%XML.Adaptor)  
|{  
|  
|}

|

Example MyClass.Person with Name and Index on Name

|

|Class MyClass.Person Extends (%Persistent, %Populate, %XML.Adaptor)  
|{

|

**| Property Name As %Name(POPSPEC = "Name()") [ Required ];  
|  
| Index NameIndex On Name;**|}

|

Example Instantiate MyClass.Person

|

| Zn "User" ;change namespace to user

| Set Oref=##class(MyClass.Person).%New()

| Set Oref.Name="Dover, Ben"

| Write Oref.%Save()

|

| ZW Oref ;ZWrite command (see Appendix E)

|

Example Display Error

|

| Set Status = Oref.%Save()

| If Status '= 1 W $SYSTEM.OBJ.DisplayError(Status)

|

Example Naming Orefs

|

| ; use descriptive Oref Names, as:

| Set PersonOref=##class(MyClass.Person).%New()

| Set ChildOref=##class(MyClass.Children).%New()

| Set CarOref=##class(MyClass.Car).%New()

| Set HouseOref=##class(MyClass.House).%New()

|

| ; not short useless names for Oref, as:

| Set Dude=##class(MyClass.Person).%New()

| Set JeffSon=##class(MyClass.Children).%New()

| Set Rod=##class(MyClass.Car).%New()

| Set Abode=##class(MyClass.House).%New()

|

Example Show Objects

|

| Write $SYSTEM.OBJ.ShowObjects()

Oref Class Name Ref Count

–––– –––––––––– –––––––––

2 MyClass.Person 1

1

|

Example Where the data is stored

|

| **Zn "User"** ;change namespace to user

| **Do ^%G**

Device:

Right margin: 80 =>

Screen size for paging (0=nopaging)? 24 =>

For help on global specifications DO HELP^%G

Global **^MyClass.PersonD**

1

^MyClass.PersonD(1)=$lb("","Dover, Ben")

Global **^MyClass.PersonI**

^MyClass.PersonI("NameIndex"," DOVER, BEN",1)=""

Example Using SQL to see your data

|

| **Zn "User**" ;change namespace to user

| **Do $SYSTEM.SQL.Shell()**

SQL Command Line Shell

–––––––––––––––––––––––––––––––––––––––––––––––––

The command prefix is currently set to:

Enter q to quit,? for help.

User>>**Select \* from MyClass.Person**

1. Select \* from MyClass.Person

ID Name

1 Dover, Ben

1 Rows(s) Affected

statement prepare time: 0.6105s, elapsed execute time: 0.0378s.

Example MyClass.Person with more Properties and Indexes

|Class MyClass.Person Extends (%Persistent, %Populate, %XML.Adaptor)  
|{  
|  
| Property Name As %Name(POPSPEC = "Name()") [ Required ];  
| Index NameIndex On Name;

|  
**| Property DOB As %Date(POPSPEC = "Date()") [ Required ];  
| Index DOBIndex On DOB;**

**|  
| Property Sex As %String(DISPLAYLIST = ",Male,Female", VALUELIST = ",M,F") [ Required ];  
| Index SexIndex On Sex;**

**|  
| Property Street As %String(POPSPEC = "Street()") [ Required ];**

**|  
| Property City As %String(POPORDER = "City()") [ Required ];  
| Index CityIndex On City;  
|  
| Property State As %String(POPSPEC = "USState()") [ Required ];  
| Index StateIndex On State;  
|  
| Property Zip As %String(POPSPEC = "USZip()") [ Required ];  
| Index ZipIndex On Zip;**|{

|

Example MyClass.Person with Method AddData

|Class MyClass.Person Extends (%Persistent, %Populate,%XML.Adaptor)  
|{  
| Property Name As %Name(POPSPEC = "Name()") [ Required ];  
| Index NameIndex On Name;

|  
| Property DOB As %Date(POPSPEC = "Date()") [ Required ];  
| Index DOBIndex On DOB;

|  
| Property Sex As %String(DISPLAYLIST = ",Male,Female", VALUELIST = ",M,F") [ Required ];  
| Index SexIndex On Sex;

|  
| Property Street As %String(POPSPEC = "Street()") [ Required ];

|  
| Property City As %String(POPORDER = "City()") [ Required ];  
| Index CityIndex On City;  
|  
| Property State As %String(POPSPEC = "State()") [ Required ];  
| Index StateIndex On State;  
|  
| Property Zip As %String(POPSPEC = "USZip()") [ Required ];  
| Index ZipIndex On Zip;  
|  
**|Method AddData(Name As %Name, DOB As %Date,**

**Sex As %String, Street As %String, City As %String,**

**State As %String, Zip As %String)**

**|{  
| Set ..Name   = Name  
| Set ..DOB    = DOB  
| Set ..Sex    = Sex  
| Set ..Street = Street  
| Set ..City   = City  
| Set ..State  = State  
| Set ..Zip    = Zip  
|}**

Example Code to run AddData Method

|

| Set Name="Volt, John"

| Set Dob=$ZDATEH("11/05/1980")

| Set Sex="M"

| Set St="100 Main"

| Set City="Evans City"

| Set State="PA"

| Set Zip="16001"

|

| Set Oref=##class(MyClass.Person).%New()

| Do Oref.AddData(Name,Dob,Sex,St,City,State,Zip)

| Set Status = Oref.%Save()

| If Status '= 1 W $SYSTEM.OBJ.DisplayError(Status)

| ZW Oref ;ZWrite command (see Appendix E)

|

Example Class Method

|

|ClassMethod HelloWorld()

|{

| Quit "Hello World"

|}

|

Example Executing a Class Method

|

| Write ##class(MyClass.Person).HelloWorld()

Hello World

|

Example Populate

|

| Write ##class(MyClass.Person).Populate(100000)

100000

|

Example Property Example

|

| Property Name As %String [ Required ];

|

Example Parameter Example

|

| Parameter MyPram as %String = "SomeValue";

|

Example Instance Method Example

|

| Method InstanceMethod(Param1 As %String) As %Status

| {

| Quit 1

| }

|

Example Class Method Example

|

|ClassMethod ClassMeth(Param1 As %String) As %Status  
|{

| Quit 1  
|}

|

Example Query Example

|

|Query MyQuery(Param1 As %String) As %SQLQuery  
|{  
| SELECT Name FROM Person  
|  WHERE (Name = :Param1)  
|}

|

Example Index Property Example

|

| Property Name As %String [ Required ];

|

| Index NameIndex on Name;

|

Example SQL Trigger Example

|

|/// This trigger updates the Log

|Trigger LogEvent [ Event = DELETE ]

|{

| Set ProcessId = $SYSTEM.SYS.ProcessID()

| Set LogOref.Entry = "Delete – "\_ProcessId

|

|}

|

Example MyClass.Parent

|

|Class MyClass.Parent Extends (%Persistent, %Populate, %XML.Adaptor, MyClass.Person)  
|{  
|  
|Property BaseballTeams As %List;  
|

|Property PlayGroups As %List ;  
|}

|

Example MyClass.Employee

|

|Class MyClass.Employee Extends (%Persistent, %Populate, %XML.Adaptor, MyClass.Person)  
|{  
|  
|Property JobTitle As %String [ Required ];  
|

|Property Company As %String [ Required ] ;  
|}

|

Example Add data to the Parent's Class

|

|Kill

|Set Name="David Crockett"

|Set Dob=$ZDATEH("11/05/1901")

|Set Sex="M"

|Set St="100 Main"

|Set City="Alamo"

|Set State="TX"

|Set Zip="15001"

|

|Set Oref=##class(MyClass.Parent).%New()

|Do Oref.AddData(Name,Dob,Sex,St,City,State,Zip)

|Set $List(BaseballTeams,1)= "Dodgers"

|Set $List(BaseballTeams,2)= "Braves"

|Set $List(BaseballTeams,3)= "Astros"

|Set $List(PlayGroups,1) = "Jan's Playgroup"

|Set $List(PlayGroups,2) = "Mike's Playgroup"

|Set Oref.BaseballTeams = BaseballTeams

|Set Oref.PlayGroups = PlayGroups

|Set Status = Oref.%Save()

|If Status '= 1 W $SYSTEM.OBJ.DisplayError(Status)

|ZW Oref ;ZWrite command (see Appendix E)

|

Example Add data to the Employee's Class

|

|Kill

|Set Name="Steve Jobs"

|Set Dob=$ZDATEH("11/05/1950")

|Set Sex="M"

|Set St="100 Main"

|Set City="Boston"

|Set State="MA"

|Set Zip="01722"

|

|Set Oref=##class(MyClass.Employee).%New()

|Do Oref.AddData(Name,Dob,Sex,St,City,State,Zip)

|Set Oref.Company = "Apple"

|Set Oref.JobTitle = "CEO"

|Set Status = Oref.%Save()

|If Status '= 1 W $SYSTEM.OBJ.DisplayError(Status)

|ZW Oref ;ZWrite command (see Appendix E)

|

Example ChildPerson Class

|

|Class MyClass.ChildPerson Extends (%Persistent, %Populate, %XML.Adaptor)  
|{

|

| Property ChildName As %String [ Required ];

|

| Property ChildDOB As %Date [ Required ];  
|  
|}

|

Example New MyClass.Person2 Class

|

|Class MyClass.Person2 Extends (%Persistent, %Populate, %XML.Adaptor)  
|{  
|  
| Property Name As %String(POPSPEC = "Name()") [ Required ];

|

| Index NameIndex On Name;  
|

| Property DOB As %Date(POPSPEC = "Date()") [ Required ];  
|  
| Index DOBIndex On DOB;  
|  
| Property Sex As %String(DISPLAYLIST = ",Male,Female", VALUELIST = ",M,F") [ Required ];  
|  
| Index SexIndex On Sex;  
|  
| Property Street As %String(POPSPEC = "Street()");  
|  
| Property City As %String(POPSPEC = "City()");

|

| Index CityIndex On City;

|

| Property State As %String(POPSPEC = "State()") [ Required ];  
|  
| Index StateIndex On State;  
|  
| Property Zip As %String(POPSPEC = "USZip()");  
|  
| Index ZipIndex On Zip;  
|   
**| Relationship LinkToChild As MyClass.ChildPerson [** **Cardinality = children, Inverse = LinkToParent ];**

|

|{

|

Example New MyClass.ChildPerson Class

|

|Class MyClass.ChildPerson Extends (%Persistent, %Populate, %XML.Adaptor)  
|{  
|  
| Property ChildName As %String(POPSPEC = "Name()") [ Required ];  
|

| Property ChildDOB As %Date [ Required ];  
|  
**| Relationship LinkToParent As MyClass.Person2 [** **Cardinality = parent, Inverse = LinkToChild ];**

|

|{

|

Example Establish a Link between a Parent Object and a Child(s) Object.

|

| Zn "User" ; change namespace to user

|

| ; these next four lines are included in case

| ; you may need to run this example again.

| ; They delete the Globals that this sequence

| ; of commands create

| Kill ^MyClass.Person2D

| Kill ^MyClass.Person2I

| Kill ^MyClass.ChildPersonD

| Kill ^MyClass.ChildPersonI

|

| ; create a new Parent Oref

| Set ParentOref=##class(MyClass.Person2).%New()

|

| ; populate the Parent Oref with data

| Set ParentOref.Name = "Ilene Dover"

| Set ParentOref.Street = "9 Goodall St."

|

| Set ParentOref.City = "Jefferson City"

| Set ParentOref.Zip = "12345"

| Set ParentOref.Sex = "F"

| Set ParentOref.State = "PA"

| ; this next line converts a date in mm/dd/yyyy

| ; format into a $H or internal Caché format

| Set ParentOref.DOB = $ZDateh("4/14/1955")

|

| ; create and populate an Oref

| ; for the first child

| Set ChildOref1=##class(MyClass.ChildPerson).%New()

| Set ChildOref1.ChildName="Larry Dover"

| ; this next line converts a date in mm/dd/yyyy

| ; format into a $H or internal Caché format

| Set ChildOref1.ChildDOB=$ZDateh("5/14/2010")

|

| ; create and populate an Oref

| ; for the second child

| Set ChildOref2=##class(MyClass.ChildPerson).%New()

| Set ChildOref2.ChildName="Moe Dover"

| ; this next line converts a date in mm/dd/yyyy

| ; format into a $H or internal Caché format

| Set ChildOref2.ChildDOB=$ZDateh("1/12/2012")

|

| ; create and populate an Oref

| ; for the third child

| Set ChildOref3=##class(MyClass.ChildPerson).%New()

| Set ChildOref3.ChildName="Curly Dover"

| ; this next line converts a date in mm/dd/yyyy

| ; format into a $H or internal Caché format

| Set ChildOref3.ChildDOB=$ZDateh("11/14/2013")

|

| ; now link the 3 children with the Parent

| ; Please note that the linking is done

| ; entirely by using just the Orefs

| Set ChildOref1.LinkToParent=ParentOref

| Set ChildOref2.LinkToParent=ParentOref

| Set ChildOref3.LinkToParent=ParentOref

|

| ; ensure the %Save call returns a one

| Set Status = ParentOref.%Save()

| If Status '=1 W $SYSTEM.OBJ.DisplayError(Status) Quit

|

| ; the following 4 calls validates that the

| ; Orefs are valid. Eqach calls should return a 1

| Write $Isobject(ParentOref)

| Write $Isobject(ChildOref1)

| Write $Isobject(ChildOref2)

| Write $Isobject(ChildOref3)

|

| ; this next call shows all objects

| ; created so far

| Write $SYSTEM.OBJ.ShowObjects()

Oref Class Name Ref Count

–––– –––––––––– –––––––––

1 MyClass.Person2 5

2 MyClass.ChildPerson 2

3 MyClass.ChildPerson 2

4 MyClass.ChildPerson 2

5 %Library.RelationshipObject 1

1

|

| ; now look at the Parent and Children Orefs,

| ; there is a lot to look at here and a lot to

| ; become confused about, but continue to look

| ; and in time it will start to make sense

| ZW ParentOref

| ZW ChildOref1

| ZW ChildOref2

| ZW ChildOref3

|

Example $ZDateh converts mm/dd/yyyy into an internal format

|

| Set ParentOref.DOB=$ZDateh("4/14/1955")

|

Example $Isobject – is it a value Object or Oref?

|

| Write $Isobject(ParentOref)

| Write $Isobject(ChildOref1)

| Write $Isobject(ChildOref2)

| Write $Isobject(ChildOref3)

|

Example $SYSTEM.OBJ.ShowObjects()

|

| Write $SYSTEM.OBJ.ShowObjects()

Example Display a Relationship with SQL

|

|DO $SYSTEM.SQL.Shell()

SQL Command Line Shell

––––––––––––––––––––––––––––––––––––––––––––––––

The command prefix is currently set to:

Enter q to quit,?for help.

User>>Select \* from MyClass.Person2

1. Select \* from MyClass.Person2

ID City DOB Name Sex State Street Zip

Jefferson City 41741 Ilene Dover F PA 9 Goodall St.12345

1 Rows(s) Affected

statement prepare time: 1.0689s, elapsed execute time: 0.0046s.

––––––––––––––––––––––––––––––––––––––––––––––––––

User>>Select \* from MyClass.ChildPerson

2. Select \* from MyClass.ChildPerson

LinkToParent ID ChildDOB ChildName

1 1||1 61860 Larry Dover 1

1 1||2 62468 Moe Dover 2

1 1||3 63140 Curly Dover 3

3 Rows(s) Affected

statement prepare time: 0.2109s, elapsed execute time: 0.0003s.

––––––––––––––––––––––––––––––––––––––––––––––––––

Example Display a Relationship from the Global(s)

|

|Do ^%G

Device:

Right margin: 80 =>

Screen size for paging (0=nopaging)? 24 =>

For help on global specifications DO HELP^%G

Global ^MyClass.Person2D

^MyClass.Person2D

1

^MyClass.Person2D(1)=$lb("","Ilene Dover",41741,"F","9 Goodall St.","Jefferson Ci

ty","PA","12345")

|

Global ^MyClass.ChildPersonD

^MyClass.ChildPersonD

3

^MyClass.ChildPersonD(1,1)=$lb("","Larry Dover",61860)

2)=$lb("","Moe Dover",62468)

3)=$lb("","CurlyDover",63140)

|

Example SQL Shell

|

| Zn "Samples" ;change namespace to Samples

| DO $SYSTEM.SQL.Shell()

| Select Name from Sample.Person

| Quit

|

Example SQL Shell showing all data elements

|

| Zn "Samples" ;change namespace to Samples

| DO $SYSTEM.SQL.Shell()

| Select \* from Sample.Person

| Quit

|

Example SQL Shell showing all data elements

|

| Zn "Samples" ;change namespace to Samples

| DO $SYSTEM.SQL.Shell()

User>>?

|

Example Type "q" to exit the SQL Shell

|

| >Zn "Samples" ;change namespace to Samples

| >DO $SYSTEM.SQL.Shell()

| >>q

| > ; out of the shell

Example Engaging the SQL Shell Log

|

| >Zn "Samples" ;change namespace to Samples

| >DO $SYSTEM.SQL.Shell()

|

| ; The log on the next line will be placed in

| ; C:\InterSystems\<Instance of Caché>

| User>>Set Log = MyLog

|

| ; Or you may specify your own directory

| User>>Set Log = C:\Users\Mike\Desktop\Mylog

|

| User>>Set Log = Off ;stops the log

|

| User>>quit ; to exit the SQL shell

| > ; now out of the shell

|

Example Embedded SQL in Caché Studio

|

|SQLRoutine

| Write !,"Running Embedded SQL"

|

| ;Host variables, pass into Embedded SQL

| New HostName

|

| &sql(Select Name into:HostName from Sample.Person)

|

| Write !,"Variable HostName: ",HostName

|

| Write !,"Embedded SQL – Exiting"

| Quit

|

|

Example Embedded SQL run from the Terminal

|

| SAMPLES>d ^SQLRoutine

| Running Embedded SQL

| Variable HostName: Adam,Emily M.

| Embedded SQL – Exiting

| SAMPLES>

|

Example Dynamic SQL

|

|SQLDynRoutine

|

| Write !,"Running Dynamic SQL"

|

| Set SqlCommand="Select Name from Sample.Person"

| Set Object=##class(%SQL.Statement).%New()

| Set StatusCode=Object.%Prepare(SqlCommand)

| If StatusCode'=1 Write StatusCode Quit

| Set Result = Object.%Execute()

| While Result.%Next()

| {Write !,Result.%Get("Name")}

|

| Write !,"Dynamic SQL – Exiting"

| Quit

|

Example Basic Class Query 1

|

|Class Sample.Query Extends Sample.Person  
|{  
|  
|Query Disp() As %SQLQuery  
|{  
| SELECT Name FROM Sample.Person  
|}

|

Example Running Basic Class Query

|

|Zn "Samples" ;change namespace to Samples

|

|Set Oref=##class(%SQL.Statement).%New()

|If '$IsObject(Oref) Write "Oref not valid object" Quit

|

|Set Status=Oref.%PrepareClassQuery("Sample.Query","Disp")

|If Status'=1 W $SYSTEM.OBJ.DisplayError(Status) Quit

|

|Set ResultSet=Oref.%Execute()

|

|While ResultSet.%Next() {Write !,ResultSet.%Get("Name")}

|

Example Basic Class Query – Stored Procedure

|

|Class Sample.Query Extends Sample.Person  
|{  
|  
|Query Disp() As %SQLQuery **[ SqlProc ]**

|  
|{  
| SELECT Name FROM Sample.Person  
|}

|

Example SQL's Custom Class Query structure

|

|Class Sample.Query2 Extends Sample.Person  
|{  
|  
| Query MyDisp(MyInput) As %Query(ROWSPEC = "Id:%String,Name:%String") [ SqlProc ]  
|{  
|}  
|  
| ClassMethod MyDispExecute(ByRef qHandle As %Binary) As %Status  
|{  
| Quit $$$OK  
|}  
|  
| ClassMethod MyDispClose(ByRef qHandle As %Binary) As %Status [ PlaceAfter = MyDispExecute ]  
|{  
| Quit $$$OK  
|}  
|  
| ClassMethod MyDispFetch(ByRef qHandle As %Binary, ByRef Row As %List,

ByRef AtEnd As %Integer = 0) As %Status

[ PlaceAfter = MyDispExecute ]  
|{  
| Quit $$$OK  
|}

|}

|

Example SQL's Custom Class Query, Selecting every tenth record

|

|Class Sample.Query2 Extends Sample.Person  
|{  
|  
|Query Disp(MyInput As %String) As %Query(ROWSPEC = "Id:%String,Name:%String") [ SqlProc ]  
|{  
|}  
|  
|ClassMethod DispExecute(ByRef qHandle As %Binary, |MyInput As %String) As %Status  
|{  
| Set qHandle=0  
| Quit $$$OK  
|}  
|  
|ClassMethod DispClose(ByRef qHandle As %Binary) As %Status [ PlaceAfter = DispExecute ]  
|{  
| Quit $$$OK  
|}  
|  
|ClassMethod DispFetch(ByRef qHandle As %Binary, |ByRef Row As %List, ByRef AtEnd As %Integer = 0) As %Status [ PlaceAfter = DispExecute ]  
|{  
**|   Set Id=qHandle  
|   Set Id=Id+10  
|   If Id > ^Sample.PersonD Set AtEnd=1,Row="" Quit $$$OK  
|   Set Oref=##class(Sample.Person).%OpenId(Id)  
|   If $IsObject(Oref) {  
|    Set Name=Oref.Name  
|    Set qHandle=Id  
|    Set Row=$LB(Id,Name)  
|   }**|   Quit $$$OK  
|}  
|}

ResultSet to run the Query

|Zn "Samples" ;change namespace to Samples

|

|Set Oref=##class(%SQL.Statement).%New()

|If '$IsObject(Oref) Write "Oref not valid object" Quit

|

|Set Status=Oref.%PrepareClassQuery("Sample.Query2","Disp")

|If Status'=1 W $SYSTEM.OBJ.DisplayError(Status) Quit

|

|Set ResultSet=Oref.%Execute()

|

|While ResultSet.%Next(){Do ResultSet.%Print() }

|

Example Tune Table Example

|

|DO $SYSTEM.SQL.TuneTable("Sample.Person",1,1)

|

Example Bitmap and Bitslice Indexes

|

| Index Prop1Index On Prop1 [ Type = bitmap ];  
|  
| Index Prop2Index On Prop2 [ Type = bitslice ];

|

Example Rebuilding Indices

|

| Do ##class(Package.Class).%BuildIndices()  
|  
| Do ##class(Package.Class)

%BuildIndices($LB("Prop1Index","Prop2Index")

|

Example

| Set File = "C:\InterSystems\TryCache\Cache.cpf"

| Write |##class(%Library.File).GetFileDateCreated(File,0)

63782,21770

|

| ;And to convert to a date humans can read:

| Write $ZDatetime(##class(%Library.File). GetFileDateCreated(File,0))

08/18/2015 06:02:50

|

Example Help Topic

|

| ; Object Help

| Do $SYSTEM.OBJ.Help()

|

| ; Version Help

| Do $SYSTEM.Version.Help()

|

| ; SQL Help

| Do $SYSTEM.SQL.Help()

|

| ; Status Help

| Do $SYSTEM.Status.Help()

|

| ; Utility Help

| Do $SYSTEM.Util.Help()

|

| ; Version Help

| Do $SYSTEM.Version.Help()

|

| ; System Help

| Do $SYSTEM.SYS.Help()

|

| ; Note: some of the following calls maybe

| ; more helpful than others. I leave that

| ; determination up to you.

|

| ; TSQL Help – Transact SQL

| Do $SYSTEM.TSQL.Help()

|

| ; Task Help

| Do $SYSTEM.Task.Help()

|

| ; Bit Help

| Do $SYSTEM.Bit.Help()

|

| ; Config Help

| Do $SYSTEM.Config.Help()

|

| ; CPU Help

| Do $SYSTEM.CPU.Help()

|

| ; CSP Help – Caché Server Pages

| Do $SYSTEM.CSP.Help()

|

| ; Dictionary Help

| Do $SYSTEM.Dictionary.Help()

|

| ; Event Help

| Do $SYSTEM.Event.Help()

|

| ; iKnow Help

| Do $SYSTEM.iKnow.Help()

|

| ; InetInfo Help – interface for the Internet

| ; address manipulation.

| Do $SYSTEM.INetInfo.Help()

|

| ; Mirror Help

| Do $SYSTEM.Mirror.Help()

|

| ; License Help

| Do $SYSTEM.License.Help()

|

| ; MV Help – MultiValue

| Do $SYSTEM.MV.Help()

|

|

Example Create a new Oref

|

| Set Oref = ##class(package.class).%New()

| Set Oref = ##class(MyClass.Person).%New()

| ZW Oref ;ZWrite command (see Appendix E)

|

| ; a second way

| Set Oref = $SYSTEM.OBJ.New("MyClass.Person")

|

| ; Note: MyClass.Person comes from **Error! Reference source not found.**

| ; *on* *Introduction to* *Classes*.

|

Example Open an existing Oref

|

| Set Oref=##class(package.class).%OpenId(Id)

|

| Set Id = 1

| Set Oref=##class(MyClass.Person).%OpenId(Id)

| ZW Oref ;ZWrite command (see Appendix E)

|

|| ; Note: MyClass.Person comes from **Error! Reference source not found.**

| ; *on* *Introduction to* *Classes*.

| ; Id of 1 must have previously been created

|

Example Save an existing Oref

|

| Set Status=Oref.%Save()

| If Status '= 1 then Write "Error"

|

Example Return a Status

|

| ; Returning a 1 or any positive number

| ; indicates a good return status

|

| ; Returning a 0 indicates an error

| ; return status

|

| Quit 1 ;good return status

|

| Do $SYSTEM.Status.OK() W 1

1

|

| Set Status=1

| If $SYSTEM.Status.IsOK(Status) W 1

1

|

| Set Status=0

| If '$SYSTEM.Status.IsOK(Status) W 0

0

|

|

Example Return a Failed Status

|

| Quit 0 ;return an error status

|

| Set Status = 0

| If $SYSTEM.Status.IsError(Status) W "Failed"

Failed

|

| Set Status = 1

| If $SYSTEM.Status.IsError(Status) W "Failed"

<>

|

Example Class Index

|

| ; combine the next two lines together

| Do ##class(%ResultSet).RunQuery

("%Dictionary.ClassDefinitionQuery","ClassIndex")

Example Summary of Classes

|

| ; combine the next two lines together

| Do ##class(%ResultSet).RunQuery

("%Dictionary.ClassDefinitionQuery","Summary")

|

Example How to tell if an ID exists

|

| ; If you don't have an Oref

| Write ##class(package.class).%ExistsId(Id)

|

| ; If you do have an Oref

| Write Oref.%ExistsId(Id)

|

Example Two ways of calling a Class Method

|

| Do ##class(Package.Class).method(params)

|

| Set Var=##class(Package.Class).method(params)

|

Example Three ways of calling a Instance Method

|

| ;First you must establish an oref, e.g.:

| ;Set oref=##Class(MyClass.Person).%New()

|

| Do oref.method(params)

|

| Write oref.method(params)

|

| Set status=oref.method(params)

|

Example Is this an Object

|

| If $Isobject(Obj) W "Is an Object"

|

| If '$Isobject(Obj) W "Not an Object"

|

Example SQL Table Listing

|

| ; the next two lines need to be combined

| ; in Terminal mode

| Do ##class(%ResultSet).RunQuery

("%Library.SQLCatalog","SQLTables")

|

Example Dump the contents of an object or Oref

|

| Set Oref=##class(MyClass.Person).%New()

| Do $SYSTEM.OBJ.Dump(Oref)

+––––––––––––––––– general information ––––––––

oref value: 1

class name: MyClass.Person

reference count: 1

+––––––––––––––––– attribute values –––––––––––

%Concurrency = 1 <Set>

City = ""

DOB = ""

Name = ""

Sex = ""

State = ""

Street = ""

Zip = ""

|

Example

| ; The Class parameter can be one class,

| ; a comma delimited list or an array

| Set Class = "MySchema.MyClass.CLS"

| ; qspec – see Flags and Qualifiers Appendix B

| Set qspec = ""

| Set fuldeploy = 0

| W $SYSTEM.OBJ.MakeClassDeployed (Class,qspec,fuldeploy)

|

Example Open a persistent object instance

|

| ; Note: MyClass.Person comes from **Error! Reference source not found.**

| ; *on* *Introduction to* *Classes*.

| Set Oref = $SYSTEM.OBJ.OpenId("Package.Class",Id)

|

| Set Oref = $SYSTEM.OBJ.OpenId("MyClass.Person",1)

|

Example Save all instances of %Library.Persistent in the process.

|

| W $SYSTEM.OBJ.SaveObjects()

|

| Set Status=$SYSTEM.OBJ.SaveObjects()

Example Return the current Object Concurrency mode.

|

| W $SYSTEM.OBJ.GetConcurrencyMode()

|

Example Set the concurrency mode for the current process/object to a new value.

|

| Set value = 3

| Set status = ""

| W $SYSTEM.OBJ.SetConcurrencyMode(value,.status)

|

Example

|

| W $SYSTEM.OBJ.GetTransactionMode()

| ; and

| W $SYSTEM.OBJ.SetTransactionMode(1,.pStatus)=1

|

Example Returns the value of an environment variable.

|

| W $SYSTEM.Util.GetEnviron("Path")

|

Example Get OSVersion Info

|

| W $SYSTEM.Util.GetOSVersionInfo()

6.2.9200

|

Example Number of CPUs

|

| W $SYSTEM.Util.NumberOfCPUs()

4

|

Example Routine Buffer Size

|

| W $SYSTEM.Util.RoutineBufferSize()

65296

|

Example

|

| W $SYSTEM.Util.RoutineBuffers(0)

0,1,0,6,0,26

| ; amount of memory, in MegaBytes, allocated for

| ; each buffer size

|

| W $SYSTEM.Util.RoutineBuffers(1)

0,430,0,430,0,430

| ; number of buffers allocated for each

| ; buffer size.

|

Example ID of the current process.

|

| W $SYSTEM.SYS.ProcessID()

3060

|

Example Compile a Class

|

| Write $SYSTEM.OBJ.Compile("Package.Class")

|

| ; Note: MyClass.Person comes from **Error! Reference source not found.**

| ; *on* *Introduction to* *Classes*.

| Write $SYSTEM.OBJ.Compile("MyClass.Person")

Compilation started on 11/27/2015 06:42:54 with qualifiers ''

Compiling class MyClass.Person

Compiling table MyClass.Person

Compiling routine MyClass.Person.1

Compilation finished successfully in 0.236s.

1

Example Compile all Classes in a Namespace

|

| Write $SYSTEM.OBJ.CompileAll()

|

Example Compile all Classes in all Namespace

|

| Write $SYSTEM.OBJ.CompileAllNamespaces()

|

Example Get information on a Compile

|

| ; Note: MyClass.Person comes from **Error! Reference source not found.**

| ; *on* *Introduction to* *Classes*.

|

| Set ver=""

| Set compiletime=""

| Set class="MyClass.Person"

|

| Do $SYSTEM.OBJ.CompileInfoClass(class,.ver,.compiletime)

| W ver

2015.2

| W compiletime

2015–12–21 05:44:22.266597

|

Example Compile from an array – first example

|

| ; Note: I included Routine1,2,3 in my array,

| ; if you do not have Routines1,2,3 then you need

| ; to create those 3 Routines. They can be simple

| ; routines with nothing in them. See

| ; **Error! Reference source not found.** *Creating a Routine*.

|

| ; first example, list is an array and called by

| ; reference (.list)

| Set errorlog=""

| Set list("Routine1.mac")=""

| Set list("Routine2.mac")=""

| Set list("Routine3.mac")=""

|

| ; qspec – see Flags and Qualifiers Appendix B

| Set qspec="/compile/displayerror/force"

|

| W $SYSTEM.OBJ.CompileList(.list,qspec,.errorlog)

Compiling routine : Routine2.MAC

Compiling routine : Routine1.MAC

Compiling routine : Routine3.MAC

Compilation finished successfully in 0.562s.

1

Example Compile from a list, define a string – second example

|

| ; second example, list is a string and

| ; called by value (no preceding ".")

|

| ; Note: I included Routine1,2,3 in my list,

| ; if you do not have Routines1,2,3 then you need

| ; to create those 3 Routines. They can be simple

| ; routines with nothing in them. See

| ; **Error! Reference source not found.** *Creating a Routine*.

|

| Set errorlog=""

| Set list="Routine1.mac,Routine2.mac,Routine3.mac"

|

| ; qspec – see Flags and Qualifiers Appendix B

| Set qspec="/compile/displayerror/force/"

| W $SYSTEM.OBJ.CompileList(list,qspec,.errorlog)

Compiling routine : Routine3.MAC

Compiling routine : Routine2.MAC

Compiling routine : Routine1.MAC

Compilation finished successfully in 0.011s.

1

Example Compile classes from a package

|

| Set package="MyClass"

|

| ; qspec – see *Flags and Qualifiers Appendix B*

| Set qspec="/compile/displayerror"

|

| Set errorlog=""

|

| W $SYSTEM.OBJ.CompilePackage(package,qspec,.errorlog)

|

Example Compile classes from a project

|

| ;You will need to create a Project in

| ;Caché Studio for this to work.

| Set project="MyProject"

|

| ; qspec – see *Flags and Qualifiers Appendix B*

| Set qspec="/compile/displayerror"

|

| Set errorlog=""

|

| W $SYSTEM.OBJ.CompileProject(project,qspec,.errorlog)

compilation started on 12/06/2015 14:08:28 with qualifiers '/compile/displayerror'

Compiling routine : Routine2.MAC

Compiling routine : Routine1.MAC

Compiling routine : Routine3.MAC

|

Example Show all supported macros defined by the system

|

| Do $SYSTEM.OBJ.ShowMacros()

|

Example Show Classes in this Namespace

|

| Do $SYSTEM.OBJ.ShowClasses()

| Do $SYSTEM.OBJ.ShowClasses("/detail")

| Do $SYSTEM.OBJ.ShowClasses("/system")

| Do $SYSTEM.OBJ.ShowClasses("/hidden")

|

Example Show Objects of this process

|

| ; qspec – see *Flags and Qualifiers Appendix B*

| Set qspec=""

| Do $SYSTEM.OBJ.ShowObjects(qspec)

|

Example Show References

|

| Do $SYSTEM.OBJ.ShowReferences(Oref,chkObj)

|

Example Show Version of the object library

|

| W $SYSTEM.OBJ.Version()

Cache Objects Version 2015.2.0.664

|

Example Is Classname valid?

|

| W $SYSTEM.OBJ.IsValidClassname("MyClass.MyName")

1

|

| ; I inserted an equal signs into the Class

| ; name which makes the name invalid

| W $SYSTEM.OBJ.IsValidClassname("MyClass.My=Name")

0

|

Example Validate indices for a class

|

| Set class = "MyClass.Person"

| Set idxList = 1 ; all indices or a $list of indexes

| Set correct = 1 ; correct any error found

| Set lock = 1 ; lock the indices while checking

|

| ; The next two lines need to be combined.

| W $SYSTEM.OBJ.ValidateIndices

(class,idxList,correct,lock)

1

|

Example Validate an Object without Saving

|

| ; MyClass.Person comes from **Error! Reference source not found.**

| ; *on* *Introduction to* *Classes*.

|

|

| ; Here we create a new Oref with no

| ; data, we should expect a lot of errors

| Set Oref=##Class(MyClass.Person).%New()

|

| ; Validate an Object (Oref) without Saving

| Set Status=Oref.%ValidateObject()

Write status ; and we do get a lot of errors

0:

|

| ; Use the $SYSTEM.Status.DecomposeStatus()

| ; function call in **Error! Reference source not found.** on System Error

| ; Calls to see the individual errors

|

Example Validate a Date

|

| Set date = $P($H,",",1) ;piece 1 of $H

| If ##class(%Library.Date).IsValid(date) W "Valid"

Valid

|

Example Validate an integer

|

| Set item=1

| If ##class(%Library.Integer).IsValid(item) W 1

1

|

| Set item=1.12

| If ##class(%Library.Integer).IsValid(item) W 1

1

|

Example Validate a numeric item

|

| Set item=1

| If ##class(%Library.Numeric).IsValid(item) W 1

1

|

| Set item="A"

| If '##class(%Library.Numeric).IsValid(item) W 0

0

|

Example Validate a Time

|

| Set item=$P($H,",",2)

|

| If ##class(%Library.Time).IsValid(item) W 1

1

|

Example Display the description of an error

|

| ; MyClass.Person comes from **Error! Reference source not found.**

| ; *on* *Introduction to* *Classes*.

|

| Set Oref=##Class(MyClass.Person).%New()

| Set Status=Oref.%Save()

|

| If Status = 1 Quit

| W $SYSTEM.OBJ.DisplayError(Status)

ERROR #5659: Property 'MyClass.Person::City(2@MyClass.Person,ID=)' required

ERROR #5659: Property

.

.

.

Example Decompose

|

| ; MyClass.Person comes from **Error! Reference source not found.**

| ; *on* *Introduction to* *Classes*.

|

| Set Oref=##Class(MyClass.Person).%New()

| Set Status=Oref.%Save()

|

| Set errorlist = ""

| Set qspec = "d"

| Set language = ""

| If Status = 1 Quit

|

| ; the next two lines need to be combined

| W $SYSTEM.Status.DecomposeStatus

(Status,.errorlist, qspec, language)

ERROR #5659: Property 'MyClass.Person::City(1@MyClass.Person,ID=)' required

ERROR #5659: Property 'MyClass.Person::DOB(1@MyClass.Person,ID=)' required

.

.

.

|

| zw errorlist

|

.

.

.

Example Delete an Object

|

| ; before you delete an Object, you first need

| ; to find its Id

|

| Set Id = 1

| Set status=##class(package.class).%DeleteId(Id)

|

Example Delete All Saved Objects

|

| Do ##class(package.class).%DeleteExtent()

|

| Do ##class(package.class).%KillExtent()

|

Example Delete all the Classes in this namespace.

|

| ; qspec – see *Flags and Qualifiers Appendix B*

| Set qspec=""

|

| W $SYSTEM.OBJ.DeleteAll(qspec)

Example Delete all classes within this package.

|

|

| Set package="MyClass"

|

| ; qspec – see *Flags and Qualifiers Appendix B*

| Set qspec=""

|

| W $SYSTEM.OBJ.DeletePackage(package.qspec)

|

Example Delete a class

|

| ; MyClass.Person comes from **Error! Reference source not found.**

| ; *on* *Introduction to* *Classes*.

| Set class="MyClass.Person"

|

| ; qspec – see *Flags and Qualifiers Appendix B*

| Set qspec=""

|

| Set errorlog=""

|

| W $SYSTEM.OBJ.Delete(class,qspec,.errorlog)

Deleting class MyClass.Person

1

Example Remove an Object (from Memory)

|

| Set Oref = ""

|

Example $SYSTEM.OBJ.Export

| ;

| ; Example – $SYSTEM.OBJ.Export

| ;

| Kill

| ;

| Set item="MyClass.Person.CLS,MyClass.ChildPerson.CLS"

| ;

| Set qspec="/displaylog/displayerror"

| ;

| Set file = "C:\Users\Mike\Desktop\ClassFile.XML"

| ;

| Set errorlog = ""

| Set Charset = ""

| ;

| W $SYSTEM.OBJ.Export(item,file,qspec,.errorlog,Charset)

Exporting to XML started on 01/27/2016 09:02:57

Exporting class: MyClass.ChildPerson

Exporting class: MyClass.Person

Export finished successfully.

1

Example $SYSTEM.OBJ.Load

| ;

| ; Example – $SYSTEM.OBJ.Load

| ;

| Kill

| ;

| Set file = "C:\Users\Mike\Desktop\ClassFile.xml"

| ;

| Set qspec = "/compile/display"

| ;

| ; unless you have a specific need

| ; for any of these parameters, leave them blank

| Set errorlog = "" ; output error log

| Set loadedlist = ""

| Set listonly = ""

| Set selecteditems = ""

| Set displayname = ""

| Set charset = ""

| Set description = ""

| ;

| ; note the parameters passed by reference

| ; the next three lines need to be combined

| Write $SYSTEM.OBJ.Load

| (file,qspec,.errorlog,.loadedlist,listonly,

| selecteditems,displayname,charset,.description)

Load started on 01/27/2016 09:06:38

Loading file C:\Users\Mike\Desktop\ClassFile.xml as xml

Imported class: MyClass.ChildPerson

Imported class: MyClass.Person, compiling 2 classes, using 4 worker jobs

Compiling class MyClass.ChildPerson

Compiling class MyClass.Person

Compiling table MyClass.ChildPerson

Compiling table MyClass.Person

Compiling routine MyClass.ChildPerson.1

Compiling routine MyClass.Person.1

Load finished successfully.

1

Example $SYSTEM.OBJ.ExportAllClasses

| ;

| ; Example – $SYSTEM.OBJ.ExportAllClasses

| ;

| Kill

| ;

| Set file = "C:\Users\Mike\Desktop\ClassFile.XML"

| ;

| Set qspec="/displaylog/displayerror"

| ;

| Set errorlog = ""

| Set Charset = ""

| ;

| ; the next two lines need to be combined

| Write $SYSTEM.OBJ.ExportAllClasses

| (file,qspec,.errorlog,Charset)

Exporting class: MyClass.ChildPerson

Exporting class: MyClass.Employee

Exporting class: MyClass.Parent

Exporting class: MyClass.Person

Exporting class: MyClass.Person2

Example $SYSTEM.OBJ.ExportAllClassesIndividual

| ;

| ; Export all classes as individual XML files into

| ; a directory.

| ;

| Kill

| ;

| Set dirname = "C:\Users\Mike\Desktop\Export"

| ;

| Set qspec="/displaylog/displayerror"

| ;

| Set errorlog = ""

| ;

| Set Charset = ""

| ;

| Set Package = "MyClass"

| ;

| Set SubDir = 1

| ;

| ; the next two lines need to be combined

| Do $SYSTEM.OBJ.ExportAllClassesIndividual

| (dirname,qspec,.errorlog,Charset,Package,SubDir)

Exporting class: MyClass.ChildPerson

Exporting class: MyClass.Employee

Exporting class: MyClass.Parent

Exporting class: MyClass.Person

Exporting class: MyClass.Person2

Exporting package: INFORMATION.SCHEMA

Example $SYSTEM.OBJ.LoadDir

| ;

| ; Example – $SYSTEM.OBJ.LoadDir

| ;

| Kill

| ;

| Set dir = "C:\Users\Mike\Desktop\Export"

| ;

| Set qspec = "/compile/display"

| ;

| Set errorlog = ""

| ;

| Set recurse = 1

| ;

| Set loadedlist = ""

| ;

| ; the next two lines need to be combined

| W $SYSTEM.OBJ.LoadDir

| (dir,qspec,.errorlog,recurse,.loadedlist)

|

Example $SYSTEM.OBJ.ExportToStream

| ;

| ; Export classes to a stream

| ;

| Kill

|

| Set item = "MyClass.Person.cls,"

| Set item = item\_"MyClass.ChildPerson.cls"

| ;

| Set stream = ##class(%Stream.GlobalCharacter).%New()

| ;

| Set qspec = ""

| ;

| Set errorlog = ""

| Set Charset = ""

| ;

| ; the next two lines need to be combined

| W $SYSTEM.OBJ.ExportToStream

| (item,stream,qspec,.errorlog,Charset)

Exporting to XML started on 01/27/2016 10:28:36

Exporting class: MyClass.ChildPerson

Exporting class: MyClass.Person

Export finished successfully.

1

Example $SYSTEM.OBJ.LoadStream

| ;

| ; Example – $SYSTEM.OBJ.LoadStream

| ;

| Kill(stream)

| ;

| Set qspec = "/compile/display"

| ;

| ; unless you have a specific need

| ; for any of these parameters, leave blank

| Set errorlog = ""

| Set loadedlist = ""

| Set listonly = ""

| Set selecteditems = ""

| Set displayname = ""

| Set charset = ""

| Set description = ""

| ;

| ; the next three lines need to be combined

| W $SYSTEM.OBJ.LoadStream

| (stream,qspec,.errorlog,.loadedlist,listonly,

| selecteditems,displayname,charset)

Load started on 01/27/2016 10:37:04

Loading file C:\InterSystems\TryCache\mgr\Temp\NNJ0f5IT2D1yaw.xml as xml

Imported class: MyClass.ChildPerson

Imported class: MyClass.Person, compiling 2 classes, using 4 worker jobs

Compiling class MyClass.ChildPerson

Compiling class MyClass.Person

Compiling table MyClass.ChildPerson

Compiling table MyClass.Person

Compiling routine MyClass.ChildPerson.1

Compiling routine MyClass.Person.1

Load finished successfully.

Example $SYSTEM.OBJ.ExportAllClassesToStream

| ;

| ;Export all Classes to a Stream

| ;

| Kill

| ;

| ; the next two lines need to be combined

| Set stream = ##class(%Stream.GlobalCharacter).%New()

| ;

| Set qspec=""

| ;

| Set errorlog = ""

| Set Charset = ""

| ;

| ; the next two lines need to be combined

| Do $SYSTEM.OBJ.ExportAllClassesToStream

(stream,qspec,.errorlog,Charset)

Exporting class: MyClass.ChildPerson

Exporting class: MyClass.Employee

Exporting class: MyClass.Parent

Exporting class: MyClass.Person

1

Example $SYSTEM.OBJ.ExportPattern

| ;

| ; Export all files matching a pattern

| ;

| Kill

| ;

| Set pattern="\*.cls"

| ;

| Set filename="C:\Users\Mike\Desktop\MyFile.txt"

| ;

| Set qspec=""

| ;

| Set errorlog=""

| Set Charset=""

| ;

| ; the next two lines need to be combined

| Do $SYSTEM.OBJ.ExportPattern

(pattern,filename,qspec,.errorlog,Charset)

Exporting class: MyClass.ChildPerson

Exporting class: MyClass.Employee

Exporting class: MyClass.Parent

Exporting class: MyClass.Person

Exporting class: MyClass.Person2

1

Example $SYSTEM.OBJ.ExportPatternToStream()

|

| ; Export all files matching a stream

| ;

| Kill

| ;

| Set pattern="\*.cls"

| ;

| ; the next two lines need to be combined

| Set stream = ##class

| (%Stream.GlobalCharacter).%New()

| ;

| Set qspec=""

| ;

| Set errorlog=""

| Set Charset=""

| ;

| ; the next two lines need to be combined

| W $SYSTEM.OBJ.ExportPatternToStream

| (pattern,stream,qspec,.errorlog,Charset)

Exporting class: MyClass.ChildPerson

Exporting class: MyClass.Employee

Exporting class: MyClass.Parent

Example $SYSTEM.OBJ.IsUpToDate()

|

| ; $SYSTEM.OBJ.IsUpToDate()

|

| Kill

|

| Set class = "MyClass.Person"

|

| Set log = 1

|

| Set type = 0

|

| Write $SYSTEM.OBJ.IsUpToDate(class,log,type)

MyClass.Person (0)

. checking recursion cache

. checking timestamps

. checking class index hash

. checking include files

. checking classtype of all the properties

. checking timestamp/hash comparison against dependency predecessors

. checking dependency predecessors recursively

MyClass.Person is up–to–date.

1

Example $SYSTEM.OBJ.Upgrade()

| ; $SYSTEM.OBJ.Upgrade()

| ;

| Kill

| ;

| Set qspec="" ;qualifiers, see *Appendix B*

| ;

| Set errorlog = ""

| ;

| Write $SYSTEM.OBJ.Upgrade(qspec,.errorlog)

No classes were modified.

1

Example $SYSTEM.OBJ.UpgradeAll()

| ; $SYSTEM.OBJ.UpgradeAll()

| ;

| Kill

| ;

| Set qspec="" ;qualifiers, see *Appendix B*

| ;

| Set errorlog = ""

| ;

| Write $SYSTEM.OBJ.UpgradeAll(qspec,.errorlog)

Example Commands verses Variables & Parameters

| ;––––––––––––––––––––––––––––––––––––––––––––––

| ; Commands, like Write or Set may be any

| ; of the following, the case of the command

| ; does not matter.

| ;––––––––––––––––––––––––––––––––––––––––––––––

| Write "this is a test"

| write "this is a test"

| Set X = "this is a test"

| seT X = "this is a test"

|

| ;––––––––––––––––––––––––––––––––––––––––––––––

| ; For Variables & Parameters however, their case

| ; must be consistent every time they are

| ; referenced

| ;––––––––––––––––––––––––––––––––––––––––––––––

| Set Var = "this is a test"

| Write Var

this is a test

|

| Set Parameter = $H

| Write $ZDateTime(Parmeter)

01/14/2016 10:53:06

| ;––––––––––––––––––––––––––––––––––––––––––––––

Example Terminal and default Namespace / Prompt

|

|USER> ;namespace and prompt "USER"

|

Example Namespaces

|

|USER> ;namespace USER

|

|SAMPLES> ;namespace SAMPLES

|

|%SYS> ;namespace %SYS

|

Example Default Namespace

|

|USER>**Write $Namespace**

USER

|

Example Changing Namespaces

|

|USER>**ZN "SAMPLES**" ;change Namespace to SAMPLES

|SAMPLES>

|SAMPLES>

|SAMPLES>**Do ^%CD** ;%CD to change to USER

Namespace: **USER**

Your in namespace USER

Default directory is c:\intersystems\trycache\mgr\user\

|USER>

Example Show all Namespaces

|

|Do ^%CD ;%CD, utility to change namespaces

|Namespace: **?** ;enter a question mark here

|

'?' for help.

'@' (at–sign) to edit the default, the last namespace

name attempted. Edit the line just as if it were

a line of code.

<RETURN> will leave you in the current

Here are the defined namespaces:

%SYS

DOCBOOK

SAMPLES

USER

|

Example Setting variable X to a value with the Set command

|

| Set X=12

| Set X="ABC"

|

Example Write command displays the value of variables

|

| Set X=12

| Write X

12

|

| Set X="ABC"

| Write X

ABC

|

Example Write command with carriage return\line feed

|

| Set X=12

| Write !,X ;carriage return\line feed

< carriage return\line feed inserted>

12

|

Example Write command displays all variables

|

| Set X=12

| Set Y=13

| Set Z=14

| Write ;Write command by itself

X=12

Y=13

Z=14

|

Example Multiple commands

|

| ;Set command with multiple variables

| Set X=12,Y=13,Z=14

|

| ;Write command with multiple variables

| Write X,!,Y,!,Z

12

13

14

|

Example Subscripted Variables

|

| ;Set subscripted variables

| Set X(1)=12

| Set X(2)=13

| Set Y(1)=14

| Set Y(2)=15

| Set Z(1,1)=16

| Set Z(1,2)=17

|

| ;Write all variables

| Write

|

X(1)=12

X(2)=13

Y(1)=14

Y(2)=15

Z(1,1)=16

Z(1,2)=17

|

Example If command with numeric operands

|

| Set X=12

| If X=12 Set X=13

| Write X

13

|

Example If command with alphanumeric operands

|

| Set X="ABC"

| If X="ABC" Set X="XYZ"

| Write X

XYZ

|

Example If command with two Variables

|

| Set X=12

| Set Y=14

| If X=12,Y=14 Set X=13

| Write X

13

|

Example Structured Code If command

|

| Set X=12

| If (X=12) {Set X=13}

| Write X

13

|

Example Structured Code If and Else commands

|

| Set X=12

| If (X=12) {Set X=13}

| Else {Write "X does not = 12"}

|

Example Structured Code If – Elseif – Else commands

|

| Set X=12

| If (X=12) {Set X=13}

| ElseIf (X=11) {Write "X = 11"}

| ElseIf (X=10) {Write "X = 10"}

| ElseIf (X=9) {Write "X = 9"}

| Else {Write "X has an unknown value"}

|

Example Plus sign, set the variable X to a value of +12

|

| Set X=+12

| Write X

12

|

Example Plus sign, set the variable X to a value of +"ABC"

|

| Set X=+"ABC"

| Write X

0

|

Example Plus sign, set the variable X to "ABC" and write +X

|

| Set X="ABC"

| Write +X

0

| Write X

ABC

|

Example Set the variable X to "ABC", if +X equals 0, write X

|

| Set X="ABC"

| If +X=0 Write X

ABC

|

Example Testing Positive and Negative numbers

|

| Set X=5

| If +X Write "True"

True

|

| Set X=100

| If +X Write "True"

True

|

| Set X=–5

| If +X Write "True"

True

|

| Set X=0

| If +X Write "True"

<>

|

Example Kill a variable

|

| Set X="ABC"

| Kill X

| Write X

<UNDEFINED>

|

Example Kill all variables

|

| Set X=12,Y=13,Z=14 ;set variables

| Write ;write all variables

X=12

Y=13

Z=14

|

| Kill ;kill all variables

| Write ;write all variables, but none exist

<>

|

Example Protect Variables from the Kill command

|

| Set X=12,Y=13,Z=14 ;set variables

| Write ;write all variables

| X=12

| Y=13

| Z=14

|

| Kill (X,Y) ;protect X and Y

|

| Write ;write all variables

X=12

Y=13

|

Example Kill a Global

|

| Set ^X="Top node of Global"

| Set ^X(1)="Subscript 1"

| Set ^X(1.5)="Subscript 1.5"

| Set ^X=(2)="Subscript 2"

| Set ^X(2.5)="Subscript 2.5"

| Set ^X(3)="Subscript 3"

| ZW ^X ;Zwrite command (see Appendix E)

^X="Top node of Global"

^X(1)="Subscript 1"

^X(1.5)="Subscript 1.5"

^X(2)="Subscript 2"

^X(2.5)="Subscript 2.5"

^X(3)="Subscript 3"

| Kill ^X

|

Example concatenating two Variables

|

| Set X="My dog's name is"

| Set Y="Teddy."

| Set DOG=X\_" "\_Y

| Write DOG

My dog's name is Teddy.

|

Example Concatenating a literal and a Variable

|

| Set Name="Fred"

| Set X="My name is: "\_Name

| Write X

My name is: Fred

|

Example Concatenating two Number

|

| Write 1\_1

11

Example Concatenation versus the plus sign

|

| ;concatenation used properly

| Write "My dog's name is "\_"Teddy."

My dog’s name is Teddy

|

| ;plus sign used improperly

| Write "My dog's name is "+"Teddy."

0

|

| ;when concatenation is used on numbers it just

| ;put the two numbers together which may not be

| ;what

| Write 1\_1

11

|

| ;plus sign used properly

| Write 1+1

2

Example Operator Precedence Comparison

|

| ;A case where precedence in Caché yields

| ;same answer as in Mathematics

| Write 5\*7+6

41

|

| ;Mathematical Operator Precedence for: 4+6\*10/5

| ;Where multiplication and division come first

| ;Is evaluated as: 4 + ((6\*10) / 5)

| ; 4 + (60 / 5)

| ; 4 + 12

| 16

|

|

| ;Caché Operator Precedence: 4+6\*10/5

| ;Evaluated strictly in a left to right order

| ; 4 + 6 \* 10 / 5

| ; 10 \* 10 / 5

| ; 100 / 5

| ; 20

|

Example Halt command

|

| Halt

|

| H

|

Example $Halt Trap Routine

|

| Set $Halt = "Stop^HaltTrap"

|

To remove the halt trap, set $Halt to a null string.

Example Remove $Halt Trap

|

| Set $Halt = ""

|

Example For Loop command – First Format

|

| For VARIABLE=START:INCREMENTAL:END CODE

|

Example For Loop command – First Format

|

| Step 1: Set the VARIABLE to the START

| Step 2: If VARIABLE is not more than END

| Execute code, otherwise stop

| Step 3: Increment the VARIABLE

| Step 4: Go to Step 2

|

Example For Loop command output

|

| For I=1:1:3 Write I,!

1

2

3

|

Example For Loop command – Second Format, First Example

|

| For VAR="VALUE1","VALUE2","VALUE3" Write !,VAR

VALUE1

VALUE2

VALUE3

|

Example For Loop command – Second Format, Second Example

|

| For VAR="VALUE1","VALUE2","VALUE3" Do Proc(VAR)

|

Example For Loop command – Third Format

|

| For Read VAR WRITE !,VAR If VAR="END" QUIT

| ;notice the two spaces after the "For"

|

Example True and False

|

| Set X=1

| If X Write "True" ;1 is always true

True

|

| Set X=0

| If X Write "True" ;0 is always false

<>

|

| Set X=""

| If X Write "True" ;null is always false

<>

|

| Set X="12abc"

| If X Write "True" ;starts with a number is true

True

|

| Set X="abc12"

| If X Write "True"

<>

Example Exercise

* Set X=1
* Set X=0
* Set X=10
* Set X=010
* Set X=50
* Set X=–10
* Set X="1"
* Set X="0"
* Set X='"0"
* Set X="1ABC"
* Set X="0ABC"
* Set X="ABC1"
* Set X=5

Example Equal Sign as a Comparison Operator

|

| Set X=1

| If X=1 Write "comparison operator"

comparison operator

|

| Set X="Now is the time"

| Write X

Now is the time

|

Example Equal Sign as a Numeric Equality Operator

|

| If 1=1 Write "True"

True

|

| Set X = 2+5

| If X = (2+5) Write "True"

True

Example Equal Sign as an Assignment Operator

|

| Set X="assignment operator"

| Write X

assignment operator

|

| Set X=1

| Write X

1

Example Not

|

| Set X=1

| Write 'X

0

|

| Set X=0

| Write 'X

1

|

| Set X="12abc"

| Set Y='X

| Write Y

0

|

| Set X="abc12"

| Set Y='X

| Write Y

1

|

| Write '5

0

|

| Write ''5

1

|

Example Exercise Not

* Set X=1
* Set X=0
* Set X='1
* Set X='0
* Set X=10
* Set X=–10
* Set X='10
* Set X='–10
* Set X="1"
* Set X="0"
* Set X='"1"
* Set X='"0"
* Set X="1ABC"
* Set X="0ABC"
* Set X='"1ABC"
* Set X='"0ABC"
* Set X="ABC1"
* Set X=5

Example Is 2 Greater Than 1?

|

| If 2>1 Write "Greater Than"

Greater Than

|

Example Is 4 Not Greater Than 4?

Or, is 4 Less Than or Equals to 4?

|

| If 4'>4 Write "Not Greater Than"

Not Greater Than

|

| If 4<=4 Write "Less Than or Equals to"

Less Than or Equals to

|

Example Is 4 less than 5?

|

| If 4<5 Write "Less Than"

Less Than

|

Example Is 5 Not Less Than 5?

|

| If 5'<5 Write "Not Less Than"

Not Less Than

|

| If 4>=4 Write "Greater Than or Equals to"

Greater Than or Equals to

|

Example – Equal To

|

| Set X=2

| Set Y=2

| If X=Y Write "Yes – X is equal to Y"

Yes – X is equal to Y"

|

| If +X=+Y Write "Yes – +X is equal to +Y"

Yes – +X is equal to +Y"

|

| Set X="ABC"

| Set Y="DEF"

| If X=Y Write "Yes – X is equal to Y"

<>

|

| If +X=+Y Write "Yes – +X is equal to +Y"

Yes – +X is equal to +Y

| ;Why?

|

|

Example *And*

|

| Set X=1

| Set Y=5

| If X&Y Write "True"

True

|

| Set X=1

| Set Y=5

| If X&&Y Write "True"

True

|

| Set X=1

| Set Y=5

| If X,Y Write "True"

| True

|

| If 1&0 Write "True"

<>

|

| Set X=1

| Set Y=5

| If X&&'Y Write "True"

<>

|

Example *Or*

|

| Set A=1

| Set B=2

| Set C=3

|

| If A=1!(B=1)!(C=1) Write "True"

True

|

| If A=1||(B=2)||(C=3) Write "True"

True

|

Example Contains

|

| If "This is our Country"["Country" Write "Yes"

Yes

|

| If "This is our Country"'["ABC" Write "No"

No

|

| If 002[2 Write "Yes"

Yes

|

Example Follows with alpha data

|

| Write $Ascii("A") ;ASCII value of "A"

65

|

| Write $Ascii("B") ;ASCII value of "B"

66

|

| If "B"]"A" Write "True" ;B follows A?

Yes

|

Example Follows Operator with numeric data

|

| Write $Ascii(2)

50

|

| Write $Ascii(1)

49

|

| If 2]19 Write "True"

True

|

Example Setting the ^TMP Global

|

| Set ^TMP("ABC")=""

| Set ^TMP(1)=""

| Set ^TMP(0)=""

| Set ^TMP(–1)=""

|

Example How a Global is sorted

|

| DO ^%G

Device:

Right margin: 80 =>

Screen size for paging (0=nopaging)? 24 =>

For help on global specifications DO HELP^%G

Global ^TMP

| ^TMP(–1)=""

| ^TMP(0)=""

| ^TMP(1)=""

| ^TMP("ABC")=""

|

Example Sort After Operator

|

| If "–1"]]"" Write "True" ;"–1" Sorts After ""

True

|

| If "1"]]"0" Write "True" ;"1" Sorts After "0"

True

|

| If "ABC"]]"1" Write "True" ;"ABC" Sorts After "1"

True

|

Example $Length function with one parameter

|

| Write $L("Page Title")

10

|

| Set X="Page Title"

| Write $L(X)

10

|

Example $Length function with two parameters

|

| Set X="One^Two^Three"

| Write $L(X,"^")

3

|

Example $Length function with two parameters demonstrated

|

| Set X="One^Two^Three"

| For I=1:1:$L(X,"^") Write $P(X,"^",I),!

One

Two

Three

|

Example $Extract function 1

|

| Set X="My dog Spot"

| Write $E(X,1,2)

My

|

Example $Extract function 2

|

| Set X="My dog Spot"

| Set $E(X,8,11)="Fred"

| Write X

My dog Fred

|

Example $Find function

|

| Set X="This is a test"

| Write $F(X,"is a")

10

|

Example $Find function

|

| Set X="This is a test"

| Write $F(X,"is a")–$L("is a")

6

|

Example $Replace function

|

| Set String="My dog is ugly"

| Set OldString="ugly"

| Set NewString="smart"

| Set String=$Replace(String,OldString,NewString)

| Write String

My dog is smart

|

Example $Translate function

|

| Set X="123,456.00"

| Set X=$TR(X,",","") ;remove commas

| Write X

123456.00

|

Example $Zconvert converts a string to Uppercase

|

| Set X="In the beginning"

| Set X=$Zconvert(X,"U")

| Write X

IN THE BEGINNING

|

Example $Zconvert converts a string to lowercase

|

| Set X="In the beginning"

| Set X=$Zconvert(X,"L")

| Write X

in the beginning

|

Example $Zconvert converts a string, all words are capitalized

|

| Set X="In the beginning"

| Set X=$Zconvert(X,"W")

| Write X

In The Beginning

|

Example $Zconvert – writes the word Caché

|

| Write "Cach"\_$Zconvert($Char(201),"L")

Caché

|

| Write $Char(201)

É

|

Example $Zstrip strips leading spaces

|

| Set X=" ABC DEF "

| Set X=$Zstrip(X,"<W")

| Write "–",X,"–"

–ABC DEF –

Example $Zstrip strips trailing spaces

|

| Set X=" ABC DEF "

| Set X=$Zstrip(X,">W")

| Write "–",X,"–"

– ABC DEF–

|

Example $Zstrip strips leading and trailing spaces

|

| Set X=" ABC DEF "

| Set X=$Zstrip(X,"<>W")

| Write "–",X,"–"

–ABC DEF–

|

Example $Zstrip strips all spaces

|

| Set X=" ABC DEF "

| Set X=$Zstrip(X,"\*W")

| Write "–",X,"–"

–ABCDEF–

|

Example $Zstrip strips a literal value

|

| Set X=" ABC 123 DEF "

| Set X=$Zstrip(X,"\*","123")

| Write "–",X,"–"

– ABC DEF –

|

Table $Data Table of Returned Values

|  |  |  |
| --- | --- | --- |
| What $Data Returns | Does the Array Node have Value? | Does the Array Node Have Descendants? |
| 0 | No | No |
| 1 | Yes | No |
| 10 | No | Yes |
| 11 | Yes | Yes |

Table 3 shows the four values returned by $Data as to the node's value and descendants.

Example Local Array

|

| Kill ;kill any variables

|

| ; this node has a value but no descendants

| Set A(1)="data"

|

| ; this node has a null value but no descendants

| Set A(2)=""

|

| ; this node has a value and by implication is a

| ; descendant of the nonexistent node A(3)

| Set A(3,1)="data"

|

| ; this node has value and (will have) descendants

| Set A(4)="data"

|

| ; this node is a descendant and (will have)

| ; descendants

| Set A(4,1)="data"

|

| ; this node is a descendant A(4,1)

| Set A(4,1,2)="data"

|

Example Array node that has a value but no descendants

|

| Write $D(A(1))

1

|

Example Array node that has a null value and no descendants

|

| Write $D(A(2))

1

|

Example Array node that has no value but has descendants

|

| Write $D(A(3))

10

|

Example Array node that has a value and descendants

|

| Write $D(A(4))

11

|

Example Array node does not exist and has no descendants, $D=0.

|

| Write $D(A(5))

0

|

Example $Get returns the variables' value

|

| Set X="ABC"

| Write $G(X)

ABC

|

Example $Get with a default parameter

|

| ; If X does not exist, then the default is used

| Kill X

| Write $G(X,"DEF")

DEF

|

Example $FNumber inserts commas in a number

|

| Write $FN(1234,",")

1,234

|

| Set X=1234

| Write $FN(X,",")

1,234

|

Example $FNumber inserts commas and a decimal point

|

| Set X=123456

| Write $FN(X,",",2) ; 2 decimal places

123,456.00

|

Example $Justify, Ten digits, right justified number

|

| Set X=123657

| Write $J(X,10)

123657

|

Example $Justify, Ten digits, with 2 decimal places

|

| Set X=123657

| Write $J(X,10,2)

123657.00

|

Example – $Justify, $FNumber, 12 Digits, 2 decimal places

|

| Set X=12345

| Write $J($FN(X,",",2),12)

12,345.00

|

Example $Justify, $FNumber, 12 digits, 2 decimal places, minus sign

|

| Set X=–12345

| Write $J($FN(X,"T,",2),12)

12,345.00–

|

Example $Ascii and $Char

|

| Write $Ascii("A")

65

| Write $Ascii("B")

66

| Write $Char(66)

B

|

Example $Char for alphabet, numbers and special characters

|

| USER>For I=33:1:126 W !,"$Char(",I,") = ",$C(I)

$Char(33) = !

$Char(34) = "

$Char(35) = #

$Char(36) = $

$Char(37) = %

$Char(38) = &

$Char(39) = '

$Char(40) = (

$Char(41) = )

$Char(42) = \*

$Char(43) = +

$Char(44) = ,

$Char(45) = –

$Char(46) = .

$Char(47) = /

$Char(48) = 0

$Char(49) = 1

$Char(50) = 2

$Char(51) = 3

$Char(52) = 4

$Char(53) = 5

$Char(54) = 6

$Char(55) = 7

$Char(56) = 8

$Char(57) = 9

$Char(58) = :

$Char(59) = ;

$Char(60) = <

$Char(61) = =

$Char(62) = >

$Char(63) = ?

$Char(64) = @

$Char(65) = A

$Char(66) = B

$Char(67) = C

$Char(68) = D

$Char(69) = E

$Char(70) = F

$Char(71) = G

$Char(72) = H

$Char(73) = I

$Char(74) = J

$Char(75) = K

$Char(76) = L

$Char(77) = M

$Char(78) = N

$Char(79) = O

$Char(80) = P

$Char(81) = Q

$Char(82) = R

$Char(83) = S

$Char(84) = T

$Char(85) = U

$Char(86) = V

$Char(87) = W

$Char(88) = X

$Char(89) = Y

$Char(90) = Z

$Char(91) = [

$Char(92) = \

$Char(93) = ]

$Char(94) = ^

$Char(95) = \_

$Char(96) = `

$Char(97) = a

$Char(98) = b

$Char(99) = c

$Char(100) = d

$Char(101) = e

$Char(102) = f

$Char(103) = g

$Char(104) = h

$Char(105) = i

$Char(106) = j

$Char(107) = k

$Char(108) = l

$Char(109) = m

$Char(110) = n

$Char(111) = o

$Char(112) = p

$Char(113) = q

$Char(114) = r

$Char(115) = s

$Char(116) = t

$Char(117) = u

$Char(118) = v

$Char(119) = w

$Char(120) = x

$Char(121) = y

$Char(122) = z

$Char(123) = {

$Char(124) = |

$Char(125) = }

$Char(126) = ~

Example $Case, First Example

|

| Write $CASE(X,1:"One",2:"Two",3:"Three",:"None")

|

| The above $Case command is equivalent to:

| For the variable X:

| If X=1 {Write "One"}

| ElseIf X=2 {Write "Two"}

| ElseIf X=3 {Write "Three"}

| Else {Write "None"}

Example $Case, Second Example

|

| Write "Input 1,2, or 3: "  
| Read X  
|  
| Do $CASE(X,1:Para1,2:Para2,3:Para3,:Error)  
| Quit  
|  
|Para1  
| Write "Processing Para1"  
| Quit  
| ;  
|Para2  
| Write "Processing Para2"  
| Quit  
| ;  
|Para3  
| Write "Processing Para3"  
| Quit

| ;  
|Error  
| Write "Processing Error"  
| Quit

|

Example $Case, third Example, this needs to be done in Caché Studio.

|

| Set DayNum=3

| Write $Case(DayNum,

| 1:"Sunday",2:"Monday",

| 3:"Tuesday",4:"Wednesday",

| 5:"Thursday",6:"Friday",

| 7:"Saturday",:"Error")

Tuesday

|

Example $Test with the Read command

|

| Read "Prompt: ",X:3 ;timed read command, 3 sec

| If $Test Write "The user answered the prompt"

| Else Write "The user did not answer"

|

Example $Increment

|

| Set ^CNTR=""

| Set X=$INCREMENT(^CNTR)

| Write ^CNTR

1

| Write X

1

|

| Set X=$INCREMENT(^CNTR)

| Write ^CNTR

2

| Write X

2

|

Example $Piece breaks down a string

|

| Kill

| Set Pets="Dog^Cat^Fish"

| Write $P(Pets,"^",1)

Dog

| Write $P(Pets,"^",2)

Cat

| Write $P(Pets,"^",3)

Fish

|

Example $Piece builds a string

|

| Set Pets=""

| Set $P(Pets,"^",1)="Dog"

| Set $P(Pets,"^",2)="Cat"

| Set $P(Pets,"^",3)="Fish"

| Write Pets

Dog^Cat^Fish

|

Example $ListBuild defines a list

|

| Set Pets=$LB("Dog","Cat","Fish")

| Write Pets

DogCatFish ; your display may be different

| ; as the invisible delimiters

| ; may display different values,

| ; this is expected behavior

Example $List returns an item from the list

|

| Set Pets=$LB("Dog","Cat","Fish")

| Write $Li(Pets,1)

Dog

| Write $Li(Pets,2)

Cat

| Write $Li(Pets,3)

Fish

|

Example Demonstrate (1) value, (2) no value and (3) undefined

|

| Kill

| Set Pets=$LB("Dog","",,"Fish")

| Write $Li(Pets,1) ;value

Dog

| Write $Li(Pets,2) ;no value

<>

| Write $Li(Pets,3) ;undefined

<NULL VALUE> ;this is an error condition

| Write $Li(Pets,4) ;value

Fish

|

Example Demonstrate (1) value, (2) no value and (3) undefined with default value

|

| Kill

| Set Pets=$LB("Dog","",,"Fish")

| Write $LG(Pets,1,"DefaultDog")

Dog

| Write $LG(Pets,2,"DefaultCat")

<>

| Write $LG(Pets,3,"DefaultTurtle")

DefaultTurtle

| Write $LG(Pets,4,"DefaultFish")

Fish

|

Example Demonstrate (1) value, (2) no value and (3) undefined

|

| Kill

| Set Pets=$LB("Dog","",,"Fish")

|

| Write $Li(Pets,1) ;value

Dog

| Write $Li(Pets,2) ;no value

<>

| Write $Li(Pets,3) ;undefined

<NULL VALUE> ;this is an error condition

| Write $Li(Pets,4) ;value

Fish

|

Example Demonstrate (1) value, (2) no value and (3) undefined with the optional var parameter

|

| Kill

| Set Pets=$LB("Dog","",,"Fish")

| Write $LD(Pets,1,var)

1

| Write var

Dog

|

| Write $LD(Pets,2,var)

1

| Write var

<>

|

| Write $LD(Pets,3,var)

0

| Write var

<>

|

| Write $LD(Pets,4,var)

1

| Write var

Fish

|

Example $ListFind Example

|

| Kill

| Set Pets=$LB("Dog","Cat","Fish")

|

| Write $LF(Pets,"Cat")

2

| Write $LF(Pets,"Snake")

0

| Write $LF(Pets,"Dog",2)

0

|

Example $ListLength returns the number of items

|

| Set Pets=$LB("Dog","Cat","Fish")

| Write $LL(Pets)

3

|

Example Cycle through the Pets List

|

| Set Pets=$LB("Dog","Cat","Fish")

| For I=1:1:$LL(Pets) {Write !,I," – ",$Li(Pets,I)}

1 – Dog

2 – Cat

3 – Fish

|

Example $ListSame compares two lists

|

| Set Pets=$LB("Dog","Cat","Fish")

| Set Pets2=$LB("Dog","Turtle","Fish")

| Set Pets3=$LB("Dog","Cat","Fish")

| Write $LS(Pets,Pets2)

0

| Write $LS(Pets,Pets3)

1

|

Example $ListSame compares two lists

|

| Set Pets=$LB("Dog","Cat","Fish")

| Set Pets2="not a valid list"

| Write $LV(Pets)

1

| Write $LV(Pets2)

0

|

Example $ListFromString creates a list from a delimited string

|

| Kill

| Set Pets1="Dog^Cat^^Fish"

| Set Pets2=$LFS(Pets1,"^") ;List From String

| Write $List(Pets2,2)

Cat

|

Example $ListToString creates a list from a delimited string

|

| Kill

| S Pets1=$LB("Dog","Cat","","Fish")

| S Pets2=$LTS(Pets1,"^") ;List From String

| W Pets2

Dog^Cat^^Fish

|

Example $ListNext sequentially returns items in a list

|

| Set Pets=$LB("Dog","Cat","","Fish")

| Set Pointer=0

| While $ListNext(Pets,Pointer,Value) {Write !,Value}

Dog

Cat

<>

Fish

|

Example $ListUpdate example

|

| Set pos = 2

| Set opt = 1 ;(1=yes, 0=no)

| Set value = "NewE12" ; new value to replace with

|

| Set MstrList = $ListBuild("El1","El2","El3")

| ; Show before or old list

| F I=1:1:3 Write $List(MstrList,I)," – "

El1 – El2 – El3 –

|

| ; Apply update

| S NewList = $ListUpdate(MstrList,pos,opt:value)

|

| ; Show after or new list

| F I=1:1:3 Write $List(NewList,I)," – "

El1 – NewEl2 – El3 -

|

Table Pattern Matching Codes

|  |  |
| --- | --- |
| Code | Meaning |
| A | Alphabetic characters |
| U | Uppercase characters |
| L | Lowercase characters |
| N | Numeric digits |
| P | Punctuation characters |
| C | Control Character |
| E | Any Character |

Table Pattern Length

|  |  |  |
| --- | --- | --- |
| Length | Meaning | |
| 4 | | exactly four |
| 1.4 | | from one to four |
| .4 | | up to four |
| 4. | | at least 4 |
| . | | any number including zero |

Example Validating Alpha Characters

|

| ;Pattern Matching "A" – alpha characters

|

| Set Data="ABCDEabcde"

| Set Pattern=".A" ;any number of alpha char

| Write Data?@Pattern ;checks for all alpha char

1 ;1=true

|

Example Validating Uppercase Characters

|

| ;Pattern Matching "U" – uppercase characters

|

| Set Data="ABCDE"

| Set Pattern=".U" ;any number of uppercase

| Write Data?@Pattern ;checks for all uppercase

1 ;1=True to the pattern

|

Example Validating a Capitalized Word

|

| ;Pattern Matching "1U.L" – Capitalize word

|

| Set Data="California" ;Data = capitalized word

| Set Pattern="1U.L" ;Pattern for capitalized

| Write Data?@Pattern

1 ;1=true is returned

|

Example Validating Numeric Digits

|

| ;Pattern Matching "N" – Numeric Digits

|

| Set Data="1234" ;Data all Numeric digits

| Set Pattern=".N" ;Pattern for numeric digit

| Write Data?@Pattern ;checks for Numeric digits

1 ;=true is returned

|

Example Validating a Numeric with Two Decimal Positions

|

| Set Data="12.34" ;data with 2 decimals

| Set Pattern=".N1"".""2N" ;Pattern

| Write Data?@Pattern

1 ;1=true is returned

|

Example Validating Punctuation Characters

|

| ;Pattern Matching "P" – Punctuation characters

|

| Set Data="., ;" ;Punctuation characters

| Set Pattern=".P" ;Pattern Punctuation

| Write Data?@Pattern ;checks for Punctuation

1 ;1=true is returned

|

Example Search a string for a substring

|

| Set String="Jack and Jill went down the hill."

| Set Pattern=".E1.P1""Jill""1P.E" ;search – “Jill”

| Write String?@Pattern

1

|

| Set String="Jack and Jill went down the hill."

| If String?.E1P1"Jill"1P.E Write "String found"

String found

|

|  |  |
| --- | --- |
| Length | Meaning |
| 2.4 | means a length of from 2 to 4 characters |
| .5 | means a length from 0 to 5 characters |
| . | means any length |
| 3 | means a length of 3 characters |

Example Pattern Matching Data of varying Lengths

|

| Set Data="12"

| If Data?1.5N Write "Numeric from 1 to 5"

Numeric from 1 to 5

|

| Set Data="54321"

| If Data?.10N Write "Numeric from 0 to 10"

Numeric from 0 to 10

|

| Set Data="ABCDE"

| If Data?1.5A Write "Alphanumeric from 1 – 5"

Alphanumeric data from 1 - 5

|

| Set Data="ABCdef123"

| If Data'?1.4A Write "Not Alphanumeric 1 – 4"

Not Alphanumeric from 1 - 4

|

Example Pattern Matching using Parenthesis or Logical “OR”

|

| Set Name="Jack"

| ;Name must be Jack or Jill or Fred

| If Name?1(1"Jack",1"Jill",1"Fred") Write 1

1

Example Pattern Matching Dates

|

| ; date format in mm/dd/yy or mm/dd/yyyy format

| Set date="05/15/2015"

| If date?1.2N1"/"1.2N1"/"2.4N Write "Valid"

Valid

|

| Set date="05/155/2015"

| If date'?1.2N1"/"1.2N1"/"2.4N Write "Invalid"

Invalid

|

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Caché Studio Keyboard Shortcuts PF1–Help | | | | | | | | | | |
| F4 | Change Namespace | Ctrl+E | Expand Cmds | | | | | Ctrl+I | | Import |
| F8 | Toggle Full Screen | Ctrl+Sh+E | Unexpand Cmds | | | | | Ctl+Sh+I | | Export |
| Navigation | | Editing | | | | Actions | | | | |
| Home | Start of Line | Shift–Del | Cut | | | Ctrl+N | | | | New |
| End | End of Line | Ctrl+C | Copy | | | Ctrl+O | | | | Open |
| Ctrl+Home | Start of I | Ctrl+V | Paste | | | Ctrl+S | | | | Save |
| Ctrl+End | End of I | Ctrl+Z | Undo | | | Ctrl+P | | | | Print |
| Ctrl+Page up | Top of Page | Ctrl+Y | Redo | | | Alt+1 | | | | Toggle Inspectr |
| Ctrl+Page down | End of Page | Ctrl+Dele | Del Next/Cur | | | Alt+2 | | | | Toggle Output |
| Ctrl+Up+Arrow | Up 1 Page | Ctrl+L | Cut Line | | | Alt+3 | | | | Toggle Worksp |
| Ctl+Dwn+Arrw | Down 1 Page | Ctrl+Alt+U | Cap Word | | | Alt+4 | | | | Toggle Watch |
| Ctrl+G | GoTo | Ctrl+Sh+U | Low Word | | | Ctrl+F7 | | | | Compile |
| Ctrl+Alt+G | Go Back | Find & Replace | | | | F7 | | | | Rebuilt All |
| Insert | | Ctrl+F | Find | | | Ctrl+Sh+V | | | | Vue Othr |
| Ctrl+[ | Insert [ ] | F3 | Find Next | | Ctrl+Alt+C | | | | Compare | |
| Ctrl+Sh+[ | Insert { } | Sh+F3 | Find Prev | | | Ctrl+w | | | | Class Browser |
| BookMarks | | Ctrl+H | Find/Repl | | | F5 | | | | Web Page |
| F2 | Next BM | Ctrl+Sh+F | Find In Files | | | Ctrl–R | | | | Reload |
| Ctrl+F2 | Toggle BMs | Debugger | | | | | | | | |
| Sh+F2 | Previous BM | Ctrl+F5 | | Start | | | Ctrl+F10 | | | Run – Cursor |
| Ctrl+Sh+F2 | Clear All BMs | Ctrl+Sh+F5 | | Restart | | | F11 | | | Step Into |
| Increase Font | Ctrl+Alt+"+" | Sh+F5 | Stop | | | | Sh+F11 | | | Step Out |
| Decrease Font | Ctrl+Alt+"–" | F9 | Toggle Break | | | | F10 | | | Step Over |

Example Your first Routine, first line

|

|JacksRoutine

|

Example Your first Routine

|

|JacksRoutine

| Write "Hello World, This is Jack!"

| Quit

|

Example Running your first Routine

|

| ZN "USER" ;change namespace to user

| Do ^JacksRoutine

Hello World. This is Jack!"

|

Example Labels and Executable Code Lines

|

|START ;Label line

| Set X=5 ;Executable code line

|

Example Do and Quit commands

|

|START

| Write !,"At Start label"

| Do PROC ;Do command

| Write !,"At first Quit"

| Quit ;Quit command

|PROC

| Write !,"At Proc label"

| Set X=5

| Write !,"At second Quit"

| Quit ;Quit command

|

Example Goto command

|

|START

| Write !,"At Start label"

| Goto PROC

| Write !,"At first Quit"

| Quit

|PROC

| Write !,"At Proc label"

| Set X=5

| Write !,"At second Quit"

| Quit

|

Example Executing a routine

|

|RTN1 ;name of the routine

| Do ^RTN2 ;execution jumps to ^RTN2 routine

| Quit

|

Example Parameter Passed by Value

|

|RTN1

| Set PARAM1="Value for Param1"

| Set PARAM2="Value for Param2"

| Do PROC^RTN2(PARAM1,PARAM2)

| Write !!,"RTN1–PARAM1: ",PARAM1

| Write !,"RTN1–PARAM2: ",PARAM2

| Quit

|

| ; --------------------------------------

|

|RTN2

|PROC(PAR1,PAR2)

| Write !,"RTN2–PAR1: ",PAR1

| Write !,"RTN2–PAR2 Old Value: ",PAR2

| Set PAR2="New value for PAR2"

| Write !,"RTN2–PAR2: New Value ",PAR2

| Quit

|

Example Parameter Passed by Reference, RTN1 and RTN2

|

|RTN1

| Set PARAM1="Value for Param1"

| Set PARAM2="Value for Param2"

| Do PROC^RTN2(.PARAM1,.PARAM2)

| Quit

|

| ; --------------------------------------

|

|RTN2

|PROC(PAR1,PAR2)

| Write !!,"RTN2 Starting"

| Write !,"RTN2–PAR1: ",PAR1

| Write !,"RTN2–PAR2: ",PAR2

| Set PAR2="New value for PAR2"

| Quit

|

Table Procedures versus Non–procedures

|  |  |
| --- | --- |
| Procedure | Non–Procedure |
| Begins and ends with curly braces {}, see next example | Begins with a Label and ends with a Quit |
| Variables created inside a Procedure are private by default and do not exist when the Procedure exits. The New command is not necessary | The New command is necessary if you wish the variable to cease to exist when the Quit is encountered. |
| Procedures are similar to Methods |  |
| Typically used in Objects style of programming | Typically used in legacy style of programming |
| Procedures may be Public or Private, default is Private | Non–Procedures are always Public |
| Input public variables must be declared | All variables are public by default |
| Scoping handled automatically, at least that is the objective | The programmer must handle the scoping manually |

Example Example of a Procedure

|

|Procedure1(Param1,Param2) [X,Y] PUBLIC {

|

| Set ABC=1 ABC is a private variable

| Set X=5 X is a public variable

|

| }

|

Example Example of a Non–Procedure

|

|NonProcedure(Param1,Param2)

|

| New XYZ ;XYZ is newed

| Set ABC=1 ;ABC is a public variable

| S XYZ="XYZ"

| Write !,"XYZ= ",XYZ  
| Write !,"Quit. XYZ will go away."

| Quit

|

Example Variables exists until the Quit command

|

|TestRtn1

| Kill ;Delete all left over variables

| Do SubProc

| ZW ;Variable B no longer exist

| Quit

|

|SubProc

| New B ;Use New when variable is created

| Set B=2

| ZW

B=2

| Quit ;Variable B is deleted on Quit

|

Example Structured Code used with If command

|

| Set X=12

| If (X=12) {Set X=13}

| Write X

13

|

|

| ;Example of non–Structured Code, the dots

| ;define the processing level

|

| If PatIns=ABC Do ;Process a Patient

| . Set PatName=^Pat("Name")

| . Set PatDob=^Pat("Dob")

| . If ^Pat("HOSP","YR")=CurrYr Do

| .. Set PatHosp=1

| .. Set PatHosYr=CurrYr

| .. ; more processing

| ; end of Patient Processing

|

|

| ;Example of Structured Code, notice the

| ;curly brackets define the processing level

| ;

| If PatIns=ABC { ;Process a Patient

| Set PatName=^Pat("Name")

| Set PatDob=^Pat("Dob")

| If ^Pat("HOSP","YR")="CurrYr {

| Set PatHosp=1

| Set PatHosYr=CurrYr

| ; more processing

| } ; end of current year

| } ; end of Patient Processing

|

|

| For Num=1:1:3 Do

| . If Num=2 Quit

| . Write !,Num

|

|

1 ;first 1 is written

3 ;then 3 is written,

| ;2 is skipped

Example For Loop command – Structured Code

|

| For Num=1:1:3 {

| If Num=2 Quit

| Write !,Num

| }

|

1 ;is the only one written

|

Example Structured Code using the Continue command

|

| For Num=1:1:3 {

| If Num=2 Continue

| Write !,Num

| }

Example Structured Code with quit command

|

|ABCRoutine

|

| Set Dummy = "Yes"

| Set X=1

| If Dummy = "Yes" {

| If X=1 {

| Write !,1

| Write !,2

| Quit ; - will quit for entire routine

| }

| }

| Quit

|

Example While command (Structured Code)

|

| Set Counter=0

| While Counter'=5 {  
| Set Counter=Counter+1  
| Write !,Counter  
| }

|

Example Do While command (Structured Code)

|

| Set Counter=0 Do {  
| Set Counter=Counter+1  
| Write !,Counter  
| } While Counter'=5

|

Example – Variable Number of Parameters

|

|PetsRoutine  
| Set X1 = "Dog"  
| Set X2 = "Cat"  
| Set X3 = "Fish"  
| Set X4 = "Pig"  
| Do CountPets(X1,X2,X3,X4)  
| Do CountPets(X1,X2)  
| Quit  
|  
|CountPets(InPets...) {  
|   Write !,"InPets is an Array, it looks like:",!  
|   ZW InPets  
|   Write !,InPets," Parameters passed in "  
|   For I=1:1:InPets {  
|    Write !," Parameter Number ",I  
|    Write " – Which is: ",InPets(I)  
| } Write !  
| }

|

Figure Output from ^PetsRoutine

|

|D ^PetsRoutine

|

InPets is an Array, it looks like:

InPets=4

InPets(1)="Dog"

InPets(2)="Cat"

InPets(3)="Fish"

InPets(4)="Pig"

4 Parameters passed in

Parameter Number 1 – Which is: Dog

Parameter Number 2 – Which is: Cat

Parameter Number 3 – Which is: Fish

Parameter Number 4 – Which is: Pig

InPets is an Array, it looks like:

InPets=2

InPets(1)="Dog"

InPets(2)="Cat"

2 Parameters passed in

Parameter Number 1 – Which is: Dog

Parameter Number 2 – Which is: Cat

|

Example Write a File

|

|WriteOutFile ;

| Set OutFile="FILE.TXT"

|

| Use 0 Write !,"Opening File FILE.TXT."

| Open OutFile:"WNS":10

| If '$Test Write "cannot open file." Quit

|

| Use OutFile

| For I=1:1:100 Write !,"Rec Num "\_I

| Close OutFile

| Use 0 Write !,"File FILE.TXT Written."

| Quit  
|

Example Read a File

|ReadInFile ;

| Set InFile="FILE.TXT"

| Close InFile

| Open InFile:"R":10

| If '$Test Write !,"cannot open file." Quit

|

| Set InCount=0

| Do $SYSTEM.Process.SetZEOF(1)

|

| Set EOF=0 Do {

| Use InFile

| Read InRecord

| If $ZEOF=–1 Set EOF=1 Quit

| Set X=$Increment(InCount)

| Use 0 Write !,InRecord

| } While EOF=0

| Use 0 Write !,InCount," Records read"

| Use 0 Write !,"End of File reached"

| Quit

|

Example Cycle through all files that matched the specifications of FILE\*.TXT

|

| CycleThruFiles ;

|

| ;Create 3 files,

| FILE1.TXT, FILE2.TXT and FILE3.TXT

|

| For File="FILE1.TXT","FILE2.TXT","FILE3.TXT" {

| Open File:"WNS"

| Close File

| }

|

| Search ;

| ;Set up for search for the files

| Set File=("FILE\*.TXT") ;use \* as a wildcard

| Set File=$ZSearch(File)

| Write !,File

|

| Do {

| Set File=$ZSearch("")

| If File="" Q

| Write !,File

| } While File'=""

|

|

Example $Now

|

| Write $Now()

63844,41064.886914

|

| Set DateNum = $PIECE($NOW(),",",1)

| Set TimeNum =$PIECE($NOW(),",",2)

| Write !,"DateNum : ",DateNum

DateNum : 63844

| Write !,"TimeNum : ",TimeNum

TimeNum : 20231.882293

|

Example $Horolog

|

| Write $H

63844, 20231

| Set DateNum = $P($H,",",1)

| Set TimeNum = $P($H,",",2)

| Write "DateNum : ",DateNum

DateNum : 63844

| Write "TimeNum : ",TimeNum

TimeNum : 20231

|

Example $ZDate

|

| Write $ZDate($H)

10/19/2015

|

Example $ZDateh

|

| Write $ZDateh("11/19/2015")

63875

|

Example $ZDatetime

|

| Write $ZDatetime($H)

10/19/2015 07:32:13

|

Example $ZDatetimeh

|

| Write $ZDatetimeh("10/19/2015 07:32:13")

63844,27133

|

Example $ZTime

|

| Write $ZTime($P($H,",",2))

10.10.04

|

Example $ZTimeh

|

| Write $ZTimeh("07:32:13")

27133

|

Example Flags and Qualifiers

|

| DO $SYSTEM.OBJ.ShowFlags()

|

| DO $SYSTEM.OBJ.ShowQualifiers()

|

Example Parameter passed to a function

|

| Set Parameter = $H

| Write $ZDateTime(Parmeter)

01/14/2016 10:53:06

|

Example MyClass.New

|

|Class MyClass.New Extends (%Persistent, %Populate, %XML.Adaptor)  
|{

|

| Parameter Param1 = 01;

**|**|}

|

Example Comment Lines

|

| Set X=1 ; this is a comment line

| Set Y=10 ; from the semicolen to end of line

| ; semicolen must start at least in column 2

|

| Set X=1 // This is a comment line

| Set Y=10 // from the slash–slash to end line

| ; slash–slash must start at least in column 2

|

Example Comment Block

|

| /\* start of a block of comment

| comments

| comments

| comments

| \*/ end of a block of comments

|

Example – Write the word Caché in Terminal

|

| Write "Cach"\_$Char(233)

|