### Основы Object/SQL

|  |
| --- |
| * **Вызвать метод класса do ##class(*package*.*class*).*method*(*arguments*) set *variable* = ##class(*package*.*class*).*method*(*arguments*) Внимание: добавьте . (точку) перед каждым параметром, передаваемым по ссылке** |
| * **Вызвать метод экземпляра do *object*.*method*(*arguments*) set *variable* = *object*.*method*(*arguments*) Внимание: добавьте . (точку) перед каждым параметром, передаваемым по ссылке** |
| * **Создать новый объект set *object* = ##class(*package.class*).%New()** |
| * **Открыть существующий объект set *object* = ##class(*package.class*).%OpenId(*id*, *concurrency*, .*status*)** |
| * **Сохранить объект set status = *object*.%Save()** |
| * **Получить ID сохранённого объекта set id = *object*.%Id()** |
| * **Получить OID сохранённого объекта set oid = *object*.%Oid()** |
| * **Определить был ли изменен объект set *variable*  = *object.*%IsModified()** |
| * **Валидировать объект без сохранения set status = *object*.%ValidateObject(),** |
| * **Валидировать свойство без сохранения set status = ##class(*package.class*).*Property*IsValid(*object.Property*)** |
| * **Вывести текст статуса ошибки do $system.Status.DisplayError(status)**   write $system.Status.GetErrorText(status) |
| * **Получить детали статуса ошибки do $system.Status.DecomposeStatus(status, .err) Внимание: err - это массив всех ошибок в статусе** |
| * **Удалить объект из памяти процесса set *object* = ""** |
| * **Удалить существующий объект класса set status = ##class(*package.class*).%DeleteId(*id*)** |
| * **Удалить все сохранённые объекты класса do ##class(*package.class*).%DeleteExtent() do ##class(*package.class*).%KillExtent()** |
| * **Перечитать из БД свойства объекта do *object*.%Reload()** |
| * **Клонировать объект set *clonedObject* = *object*.%ConstructClone()** |
| * **Вывести или задать значение свойства write *object*.*property* set *object*.*property* = *value*** |
| * **Вывести параметр класса write ##class(*package.class).*#*PARAMETER*** |
| * **Задать сериализованное (встроенное) свойство set *object*.*property.embeddedProperty* = *value*** |
| * **Связать два объекта set *object1*.*referenceProperty* = *object2*** |
| * **Заполнить класс do ##class(*package*.*class*).Populate(*count*, *verbose*)** |
| * Удалить все объекты из памяти процесса kill |
| * Перечислить все объекты в памяти процесса do $system.OBJ.ShowObjects() |
| * **Отобразить все свойства объекта do $system.OBJ.Dump(*object*)**   **zwrite *object* (v2012.2+)** |
| * **Определить, является ли значение объектной ссылкой if $isobject(*variable*) Возвращает 1=да, 0=нет** |
| * **Найти имя класса объекта write $classname(*oref*)** |
| * Запустить оболочку SQL do $system.SQL.Shell() |
| * **Протестировать запрос класса do ##class(%ResultSet).RunQuery(*class*, *query*)** |
| * **Объявить тип переменных для помощника Студии #dim *object* as *package.class*** |

### Команды ObjectScript

|  |
| --- |
| * **Continue Прекратить текущую итерацию цикла, продолжить цикл.** |
| * **Do Выполнить метод, процедуру или программу.** |
| * **For {}, While {}, Do {} While Несколько раз выполнить блок кода.** |
| * **Halt Остановить процесс Caché и закрыть Терминал.** |
| * If {} ElseIf {} Else {} Оценка условия и ветвление. |
| * **Quit, Return (v2013.1+) Завершить метод, процедуру или программу. Опционально вернуть значение вызывающему методу. Завершение циклов.** |
| * Set Установить значение переменной. |
| * Try {} Catch {}, Throw Обработать ошибки. |
| * Write Отобразить текстовые строки, значение переменной или выражения. |
| * **ZWrite Отобразить массивы, списки строк, битовые строки.** |
| * Kill Уничтожить переменные. |

### Функции ObjectScript для работы с датой/временем и специальными переменными

|  |
| --- |
| * **Конвертация даты (внутренний  внешний) $zdate(*internalDate, format*)** |
| * **Конвертация даты (внешний  внутренний) $zdateh(“*mm/dd/yyyy*”)** |
| * **Конвертация времени (внутренний  внешний) $ztime(*internalTime, format*)** |
| * **Конвертация времени (внешний  внутренний) $ztimeh(“*hh:mm:ss*”)** |
| * **Текущая строка с локальной датой/временем $horolog** |
| * **Текущая строка с UTC датой/временем $ztimestamp** |

### Функции разветвления в ObjectScript

|  |
| --- |
| * **Вернуть результат по значению выражения $case(*expression*, *value1*:*result1*, *value2*:*result2*, …, :*result)*** |
| * **Вернуть результат для первого истинного условия $select(*condition1*:*result1*, *condition2*:*result2*, …, 1:*resultN*)** |

### Функции для работы со строками в ObjectScript

|  |
| --- |
| * **Вычленить символы из строки$extract(*string, start, end*)** |
| * **Выровнять по правому краю шириной *width* $justify(*string, width*)** |
| * **Получить длину строки $length(*string*)** |
| * **Получить количество подстрок в строке $length(*string, delimiter*)** |
| * **Построить список set *listString* = $listbuild(*list items, через запятую)*** |
| * **Вставить подстроку с список set $list(*listString, position*) = *substring*** |
| * **Получить элемент из списка set *variable* = $list(*listString, position*)** |
| * **Получить количества подстрок в списке $listlength(*listString*)** |
| * **Получить подстроку из строки с разделителем $piece(*string*, *delimiter, pieceNumber)*** |
| * **Вставить подстроку в список с разделителем set $piece(*string*, *delimiter, pieceNumber) = piece*** |
| * **Заменить/удалить подстроку из строки $replace(*string, searchString, replaceString*)** |
| * **Получить строку в обратном порядке $reverse(*string*)** |
| * **Заменить/удалить символы из строки $translate(*string, searchChars, replaceChars*)** |

### Функции для проверки существования в ObjectScript

|  |
| --- |
| * **Проверить, существует ли переменная $data(*variable*)** |
| * **Вернуть значение существующей переменной или значение по умолчанию $get(*variable*)** |
| * **Вернуть следующий индекс в разреженном массиве $order(*array*(*subscript*))** |

### Специальные переменные в ObjectScript

|  |
| --- |
| * **Инкременитровать глобал $increment(^global, increment) $sequence(^global, increment)** |
| * **Сравнить с регулярным выражением $match(*string*, *regularexpression*)** |
| * **Отобразить случайное число $random(*count*) + *start*** |

### Дополнительные функции ObjectScript

|  |
| --- |
| * **ID процесса $job** |
| * **Текущая область имен $namespace** |
| * **Роли безопасности $roles** |
| * **Имя пользователя безопасности $username** |

### Полезные команды

|  |
| --- |
| * Изменить область do ^%CD   **znspace "*namespace*"** |
| * **Отобразить глобал do ^%G zwrite *global*** |

### Коллекции-списки

|  |
| --- |
| * **Создать новый отдельный список set *listObject*=##class(%ListOfDataTypes).%New()** |
| * Работа со свойством списка Используйте методы ниже на свойстве коллекции-списке |
| * **Вставить элемент в конец списка do *listObject*.Insert(*value*) do *object.listProperty.*Insert(*value*)** |
| * **Вставить элемент в список do *listObject*.SetAt(*value, position*) do *object.listProperty.*SetAt(*value, position*)** |
| * **Удалить элемент из списка do *listObject.*RemoveAt(*position*) do *object.listProperty*.RemoveAt(*position*)** |
| * **Получить элемент списка set *variable = listObject*.GetAt(*position*) set *variable* = *object.listProperty.*GetAt(*position*)** |
| * **Получить размер списка set *variable* = *listObject*.Count() set *variable* = *object.listProperty*.Count()** |
| * **Удалить все элементы из списка do *listObject*.Clear() do *object.listProperty.*Clear()** |

### Коллекции-массивы

|  |
| --- |
| * **Создать новый отдельный массив set *arrayObject*=##class(%ArrayOfDataTypes).%New()** |
| * Работа со свойством массива Используйте методы ниже на свойстве коллекции-массиве |
| * **Вставить элемент в массив do *arrayObject*.SetAt(*value*, *key*) do *object*.*arrayProperty*.SetAt(*value*, *key*)** |
| * **Удалить элемент из массива do *arrayObject.*RemoveAt(*key*) do *object.arrayProperty*.RemoveAt(*key*)** |
| * **Получить элемент массива set *variable* = *arrayObject*.GetAt(*key*) set *variable* = *object.arrayProperty*.GetAt(*key*)** |
| * **Получить размер массива set *variable* = *arrayObject*.Count() set *variable* = *object.arrayProperty*.Count()** |
| * **Удалить все элементы из массива do *arrayObject*.Clear() do *object.arrayProperty*.Clear()** |

### Отношения

|  |
| --- |
| * **Создание ссылки "родитель-к-потомкам" do *parentObject*.*childRefProperty.*Insert(*childObject*) set *childObject.parentRefProperty* = *parentObject*** |
| * **Создание ссылки "один-ко-многим" do *oneObject*.*manyRefProperty.*Insert(*manyObject*) set *manyObject.OneRefProperty* = *OneObject*** |
| * **Получить свойство объекта-потомка set *variable* = *parentObject*.*childRefProperty.*GetAt(*position*).*property*** |
| * **Получить свойство объекта "многие" set *variable* = *oneObject*.*manyRefProperty.*GetAt(*position*).*property*** |
| * **Получить количество объектов потомок/многие set *variable* = *parentObject*.*childRefProperty.*Count() set *variable* = *oneObject.manyRefProperty.*Count()** |
| * **Открыть объект потомок/многие напрямую set *object* = ##class(*package.class*).IDKEYOpen(*parentID, childsub*)** |
| * **Получить id объекта-потомка set status = ##class(*package.class*).IDKEYExists(*parentID, childsub, .childID)*** |
| * **Удалить объекты типа потомок/многие do *parentObject*.*childRefProperty.*Clear() do *oneObject.manyRefProperty.*Clear()** |

### Потоки

|  |
| --- |
| * **Создать поток set *streamObject*=##class(%Stream.GlobalCharacter).%New() set *streamObject*=##class(%Stream.GlobalBinary).%New() или использовать методы ниже на свойстве потока** |
| * **Добавить текст в поток do *streamObject*.Write(*text*) do *object.streamProperty.*Write(*text*)** |
| * **Добавить строку текста в поток do *streamObject*.WriteLine(*text*) do *object.streamProperty.*WriteLine(*text*)** |
| * **Читать *len* символов текста из потока write *streamObject.*Read(*len*) write *object*.*streamProperty*.Read(*len*)** |
| * **Читать строку текста из потока write *streamObject*.ReadLine(*len*) write *object*.*streamProperty*.ReadLine(*len*)** |
| * **Перейти в начало потока do *streamObject*.Rewind() do *object.streamProperty*.Rewind()** |
| * **Перейти в конец потока для добавления do *streamObject*.MoveToEnd() do *object.streamProperty*.MoveToEnd()** |
| * **Очистить поток do *streamObject*.Clear() do *object.streamProperty*.Clear()** |
| * **Отобразить длину потока write *streamObject*.Size write *object*.streamProperty.Size** |

### Макросы юнит-теста

|  |
| --- |
| * **Утверждение равенства do $$$AssertEquals(*value1, value2, message)*** |
| * **Утверждение неравенства do $$$AssertNotEquals(*value1, value2, message)*** |
| * **Утверждение статуса OK do $$$AssertStatusOK(*status, message)*** |
| * **Утверждение статуса не OK do $$$AssertStatusNotOK(*status, message)*** |
| * **Утверждение условия = true do $$$AssertTrue(*condition, message)*** |
| * **Утверждение условия не равно true do $$$AssertNotTrue(*condition, message)*** |
| * **Логировать сообщение do $$$LogMessage(*message)*** |

### Другие макросы

|  |
| --- |
| * **Вернуть статус OK quit $$$OK** |
| * **Вернуть статус ошибки quit $$$ERROR($$$GeneralError, *message*)** |
| * **Проверить, хорош ли статус if $$$ISOK(*status*)** |
| * **Проверить, ошибочен ли статус if $$$ISERR(*status*)** |
| * Вернуть нулевую ссылку объекта quit $$$NULLOREF |
| * **Поместить разрыв (новую строку) в строке write *string1\_*$$$NL\_*string2*** |



