Artículo
Kurro Lopez · 30 ago, 2019 Lectura de 4 min

Métodos autogenerados de gran utilidad

¡Hola a tod@s!

Para cada propiedad, consulta o índice definido, se generarán automáticamente varios métodos correspondientes en una compilación de clase. Estos métodos pueden ser muy útiles. En este artículo, describiré algunos de ellos.

Propiedades

Supongamos que ha definido una propiedad llamada "Property". Los siguientes métodos quedarán disponibles automáticamente (la propiedad en negrita es una parte variable, igual al nombre de la propiedad):

```
ClassMethod PropertyGetStored(id)
```

Para las propiedades de tipo de datos, este método devuelve su valor lógico. Para propiedades de objetos, devuelve su id. Es una referencia global envuelta a la variable global de datos de clase, y la forma más rápida de recuperar el valor de la propiedad única. Este método solo está disponible para propiedades almacenadas.

```
Method PropertyGet()
```

Esto sería un método para recuperar el valor de la propiedad.

```
Method PropertySet(val) As %Status
```

Esto sería un método para asignar el valor de la propiedad.

Propiedades de objetos

Si es una propiedad de objeto, tendrá disponibles algunos métodos adicionales, relacionados con el acceso ID y OID:

```
Method PropertySetObjectId(id)
```

Este método define el valor el valor de la propiedad por ID, por lo que no será necesario abrir un objeto para definirlo como valor de una propiedad.

```
Method PropertyGetObjectId()
```

Este método devuelve el ID del valor de la propiedad.

```
Method PropertySetObject(oid)
```

Este método define el valor de propiedad por OID.

```
Method PropertyGetObject()
```

Este método devuelve el OID del valor de la propiedad.

Propiedades de tipos de datos

Para una propiedad de tipo de datos, hay disponibles varios otros métodos para convertir entre distintos formatos:

```
ClassMethod PropertyDisplayToLogical(val)
ClassMethod PropertyLogicalToDisplay(val)
ClassMethod PropertyOdbcToLogical(val)
ClassMethod PropertyLogicalToOdbc(val)
ClassMethod PropertyXSDToLogical(val)
ClassMethod PropertyLogicalToXSD(val)
```

```
ClassMethod PropertyIsValid(val) As %Status
```

Verifica si *val* es un valor de propiedad válido

```
ClassMethod PropertyNormalize(val)
```

Devuelve un valor lógico normalizado

Notas

- Las relaciones son propiedades y se pueden definir y recuperar (get/set) con estos métodos
- El *val* de entrada siempre es un valor lógico, excepto para los métodos de conversión de formato.

Índices

Para un índice llamado "**Index**", los siguientes métodos estarán disponibles automáticamente

```
ClassMethod IndexExists(val) As %Boolean
```

Devuelve 1 o 0 dependiendo de si existe un objeto con este *val*, donde *val* es un valor lógico de la propiedad indexada.

Índices únicos

Para índices únicos habrá métodos adicionales disponibles:

```
ClassMethod IndexExists(val, Output id) As %Boolean
```

Devuelve 1 o 0 dependiendo de si existe un objeto con este *val*, donde *val* es un valor lógico de la propiedad indexada. También devuelve el *id* del objeto (si se encontrara) como segundo argumento.

```
ClassMethod IndexDelete(val, concurrency = -1) As %Status
```

Elimina la entrada cuyo valor de índice igual a *val*.

```
ClassMethod IndexOpen(val, concurrency, sc As %Status)
```

Devuelve un objeto existente con valor de índice igual a *val*.

Notas**:**

a) Como un índice puede basarse en varias propiedades, esta firma de método cambiaría para recibir varios valores como entrada, como por ejemplo en el siguiente índice:

```
Index MyIndex On (Prop1, Prop2);
```

Entonces el método IndexExists tendría la siguiente firma:

```
ClassMethod IndexExists(val1, val2) As %Boolean
```

Donde *val1* corresponde al valor *Prop1* y *val2* corresponden al valor *Prop2*. Otros métodos siguen la misma lógica.

b) Caché genera un índice IDKEY que indexa el campo ID (RowID). El usuario puede redefinirlo y también puede contener varias propiedades. Por ejemplo, para verificar si la clase tiene alguna propiedad definida, ejecute:

```
Write ##class(%Dictionary.PropertyDefinition).IDKEYExists(class, property)
```

c) Todos los métodos de índices buscan un valor lógico

d) Documentación

Consultas

Para una consulta (que puede ser una simple consulta SQL o una consulta de clase personalizada, aquí está mi publicación sobre las mismas) llamada "Query", se genera el método Func:

```
ClassMethod QueryFunc(Arg1, Arg2) As %SQL.StatementResult
```

que devuelve un objeto %SQL.StatementResult usado para iterar sobre la consulta. Por ejemplo, la clase Sample.Person en el namespace Samples tiene una consulta ByName que acepta un parámetro. Se puede llamar desde el contexto de un objeto con este código:

```
Set ResultSet=##class(Sample.Person).ByNameFunc("A")
While ResultSet.%Next() { Write ResultSet.Name,! }
```

Además, también hay una clase demo en GitHub que muestra el funcionamiento de estos métodos.

#Code Snippet #Modelo de datos de objetos #Caché #InterSystems IRIS