Article
Irène Mykhailova · Juin 7, 2022 4m de lecture

Méthodes utiles générées automatiquement

Pour chaque propriété, requête ou index défini, plusieurs méthodes correspondantes seraient automatiquement générées lors de la compilation d'une classe. Ces méthodes peuvent être très utiles. Dans cet article, je décrirai certaines d'entre elles.

Properties

Disons que vous avez défini une propriété nommée "Property". Les méthodes suivantes seraient automatiquement disponibles (la propriété indiquée en gras est une partie variable, égale au nom de la propriété) :

```
ClassMethod PropertyGetStored(id)
```

Pour les propriétés de type de données, cette méthode renvoie leur valeur logique, pour les propriétés d'objet, elle renvoie l'id. C'est une référence globale wrappée à la globale de données de la classe et le moyen le plus rapide de récupérer la valeur de la propriété singulière. Cette méthode n'est disponible que pour les propriétés stockées.

```
Method PropertyGet()
```

C'est un getter de propriété. Peut être redéfini.

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

C'est un définisseur de propriété. Peut être redéfini.

Propriétés de l'objet

S'il s'agit d'une propriété objet, certaines méthodes supplémentaires, liées à l'accès aux ID et OID, deviennent disponibles :

```
Method PropertySetObjectId(id)
```

Cette méthode définit la valeur de la propriété par ID, il n'est donc pas nécessaire d'ouvrir un objet pour le définir comme valeur de la propriété.

```
Method PropertyGetObjectId()
```

Cette méthode renvoie l'ID de la valeur de la propriété.

```
Method PropertySetObject(oid)
```

Cette méthode définit la valeur de la propriété par OID.

```
Method PropertyGetObject()
```

Cette méthode renvoie l'OID de la valeur de la propriété.

Propriétés du type de données

Pour une propriété de type de données, plusieurs autres méthodes de conversion entre différents formats sont disponibles :

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

ClassMethod PropertyIsValid(val) As %Status
```

Vérification de la validité de la valeur de la propriété

```
ClassMethod PropertyNormalize(val)
```

Renvoi de la valeur logique normalisée

Remarques

- Les relations constituent des propriétés et peuvent être obtenues ou définies à l'aide des méthodes suivantes
- L'entrée val est toujours une valeur logique, sauf pour les méthodes de conversion de format.

Index

Pour un index nommé "Index", les méthodes suivantes seraient automatiquement disponibles

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

Renvoi de 1 ou 0 selon l'existence d'un objet avec ce val, où val est une valeur logique de la propriété indexée.

Index uniques

Pour les index uniques, des méthodes supplémentaires sont disponibles :

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

Renvoi de 1 ou 0 en fonction de l'existence d'un objet avec ce val, où val est une valeur logique de la propriété indexée. Renvoi également de l'identifiant de l'objet (s'il a été trouvé) comme second argument.

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

Suppression de l'entrée dont la valeur d'index est égale à val.

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

Renvoi de l'objet existant dont l'indice est égal à val.

Remarques:

a) Étant donné qu'un index peut être basé sur plusieurs propriétés, la signature de la méthode serait modifiée pour avoir plusieurs valeurs en entrée, par exemple, considérez cet index :

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

La méthode IndexExists aurait alors la signature suivante :

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

Où val1 correspond à la valeur de Prop1 et val2 correspond à la valeur de Prop2. Les autres méthodes suivent la même logique.

b) Caché génère un index IDKEY qui indexe le champ ID (RowID). Il peut être redéfini par l'utilisateur et peut également contenir plusieurs propriétés. Par exemple, pour vérifier si une classe a une propriété définie, exécutez :

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

c) Toutes les méthodes d'indexation vérifient la présence d'une valeur logique

d) Documentation

Requêtes

En ce qui concerne une requête (qui peut être une simple requête SQL ou une requête de classe personnalisée, voici mon post à ce sujet) nommée "Query", la méthode Func est générée :

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

qui renvoie un %SQL.StatementResult utilisé pour itérer sur la requête. Par exemple, la classe Sample.Person de l'espace de noms Samples possède une requête ByName acceptant un paramètre. Elle peut être appelée depuis le contexte objet au moyen de ce code :

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

En outre, une classe de démonstration sur GitHub illustre ces méthodes.

#Bonnes pratiques #Code Snippet #Object Data Model #Caché #InterSystems IRIS

URL de la

source:https://fr.community.intersystems.com/post/m%C3%A9thodes-utiles-g%C3%A9n%C3%A9r%C3%A9es-automatiquement