IKVMJavaLibrary vs CLILibrary

domenica 21 novembre 2010
11:47

Intro

Mostro alcuni esempi di utilizzo di oggetti .NET sia utilizzando la CLILibrary (.NET) che la JavaLibrary compilata da IKVM, quindi versione scritta in Java ma tradotta per .NET.
Questo lavoro mi servirà per capire dove devo andare ad inserire le convenzioni e il risultato che dovrò ottenere alla fine.

Glossario

IKVMJavaLibrary: versione della JavaLibrary compilata per la piattaforma .NET con l'utilizzo di IKVM.
CSStudent.Student: classe scritta in C# per fare qualche esempio

Creazione Oggetti

IKVMJavaLibrary	CLILibrary
java_object('CStudent.Student, CStudent',[311471,'ale','monta'], Obj),	cli_object(CSConvention,'CStudent.dll','CStudent.Student',[311471,'ale','monta'],Obj),

La differenza sta nel fatto che con la JavaLibrary compilata con IKVM occorre specificare il FullyName della classe nel formato .NET, eventualmente anche con la specifica di versione, cultura e chiave pubblica.
Il predicato java_object andrebbe esteso per accettare la convenzione da associare all'oggetto che si sta creando.

Metodi

IKVMJavaLibrary	CLILibrary
Obj <- 'PrintStudent' returns StudStr,	Obj <- printStudent returns StudStr,

L'unica differenza ovviamente sta nel fatto che la CLILibrary utilizzando le convenzioni permette di scrivere il nome del metodo anche con la minuscola.

Proprietà / Eventi

IKVMJavaLibrary	CLILibrary
Obj <- 'get_Name' returns Name,	Obj.name <- get(Name),
Obj <- 'set_Name'('ciaoProp'),	Obj.name <- set('ciaoProp'),

Qui c'è la differenza più sostanziale.
Infatti la JavaLibrary è sviluppata in Java dove non esiste il concetto di Property ma solo di Campi e Metodi.
Però fortunatamente il compilatore di Visual Studio compila le Property in metodi, quindi specificando il nome del metodo opportuno riesco ad accedere alle property anche attraverso la JavaLibrary.
Però specificare il nome esatto del metodo "non è una bella cosa" (potrei non conoscerlo), quindi qui bisognerà effettuare delle conversioni in base alle Convenzioni.

Inoltre si potrebbe perdere un po' di espressività: cioè con la CLILibrary per accedere alle Property utilizzo una sintassi del tutto simile alla sintassi di un qualunque linguaggio .NET (Oggetto.Property), mentre con la JavaLibrary compilata con IKVM è come se invocassi un metodo (<-).

Campo Pubblico

IKVMJavaLibrary	CLILibrary
Obj.publicField <- get(Field),	Obj.publicField <- get(Field),
Obj.publicField <- set('ciao'),	Obj.publicField <- set('ciao'),

Come si può notare in questo caso non ci sono differenze oltre alle eventuali conversioni da fare sul nome del campo.

Metodo Statico

IKVMJavaLibrary	CLILibrary
class('CStudent.Student, CStudent') <- 'PrintInfoUni' returns Info,	class('CStudent.Student, CStudent') <- printInfoUni returns Info,

L'unica differenza sta nell'utilizzo delle convenzioni nella CLILibrary.

Proprietà statica

IKVMJavaLibrary	CLILibrary
class('CStudent.Student, CStudent') <- 'get_StaticProperty' returns StaticProp,	class('CStudent.Student, CStudent').staticProperty <- get(StaticField2),
class('CStudent.Student, CStudent') <- 'set_StaticProperty'('MyStaticProp'),	class('CStudent.Student, CStudent').staticProperty <- set('CIOAIAO').

Stesse differenze che ci sono per le Property di istanza.

Campo pubblico statico

IKVMJavaLibrary	CLILibrary
class('CStudent.Student, CStudent').publicStaticField <- get(StaticField),	class('CStudent.Student, CStudent').publicStaticField <- get(StaticField),
class('CStudent.Student, CStudent').publicStaticField <- set('MyStaticString'),	class('CStudent.Student, CStudent').publicStaticField <- set('MyStaticString'),

Nessuna differenza oltre alle convenzioni.

Considerazioni

Questi esempi sono stati fatti su un oggetto C#, comunque la storia non cambia: ogni volta che si cerca di invocare un metodo, accedere ad una property o ad un campo occorre modificarne il nome in base alla convenzione a cui è associato quell'oggetto.