The REBOL Documentation Project

-- FR - Documentation REBOL - Articles Techniques --

Articles Techniques

CATCH existe en REBOL

REBOLtof

Première publication : 11 août 2005, et mis en ligne le jeudi 11 août 2005

Résumé :

Où cela ? En fait, il en existe deux : la fonction CATCH et l'attribut de fonction [catch]

Où cela ? En fait, il en existe deux : la fonction CATCH et l'attribut de fonction [catch]

Version : 1.1
Date : 11-aug-05
Auteur : Ladislav Mecir
Traducteur : Christophe "REBOLtof" Coussement

- 1 Historique
- 2 La fonction CATCH
- 3 L'attribut de fonction [catch]
- 4 Comparaison
- 5 Interférence
- 6 Solution

1 Historique

Date	Version	Commentaires	Auteur	Email
11-aug-05	1.0	initial	REBOLtof	reboltof-at-yahoo-dot-com
11-aug-05	1.1	ajout de détails et corrections à la demande de l'auteur	REBOLtof	reboltof-at-yahoo-dot-com

2 La fonction CATCH

Cette fonction nous permet principalement de forcer la fin d'une évaluation compliquée.

Exemple

```
catch [
    ; une évaluation compliquée...
    if done? [throw result1]
    ; encore des évaluations...
    if done? [throw result2]
    ; d'autres évaluations...
    ; maintenant nous avons terminé!
    result3
]
```

Un résultat similaire peut être obtenu en utilisant ANY, ALL ou CASE. Cependant, CATCH sera en général le plus approprié pour ce genre de travail.

Si nous utilisons une fonction CATCH "nidifiée", nous pouvons avoir recours au raffinement CATCH/NAME afin d'être en mesure de spécifier la "cible" désirée pour accueillir les résultats :

```
catch/name [
    ; ...
```

```
    catch/name [
        ; ...
        if done-a? [throw/name result 'a] ; pour le 'a du CATCH (le CATCH "externe")
        ; ...
        if done-b? [throw/name result 'b] ; pour le 'b du CATCH (le CATCH "interne")
        ; ...
        result-b
    ] 'b
    ; ...
    if done-a? [throw/name result 'a]
    ; ...
    result-a
] 'a
```

3 L'attribut de fonction [catch]

Cet attribut est utilisé afin de permettre un traitement de l'erreur dans les fonctions, en dehors de la procédure normale.

Exemple : supposons que nous avons besoin d'une fonction HANDLE afin d'appliquer un traitement sur tous les nombres, sauf sur 66. Nous pouvons considérer que c'est une erreur d'appeler la fonction en lui passant un argument égal à 66...

```
handle: func [
    [catch]
    argument1 [integer!]
    argument2 [block!]
] [
    if argument1 = 66 [throw make error! "Le nombre 66 n'est pas autorisé comme argument !"]
    do argument2
]
```

L'expression :

```
handle 1 ["OK"]
```

Donne "OK", tandis que

```
handle 66 ["OK"]
```

déclenche une erreur.

4 Comparaison

- L'attribut de fonction [catch] a un objectif différent que la fonction CATCH.
- Il n'existe pas de mécanisme identique afin comprendre la signification de THROW pour les

fonctions nidifiées (NdT : il n'existe pas de raffinement /NAME similaire à celui de la fonction CATCH).

5 Interférence

Un attribut de fonction [catch] nidifié n'est pas en mesure de distinguer la signification de THROWs, et "intercepte" tous les THROWs pour autant qu'ils soient des erreurs.

Exemple :

```
index: 1
block: copy ["OK"]
insert tail block make error! "error"
type? catch [
    ; ...
    handle 1 [throw pick block index]
]
```

Cela est correct jusque maintenant, mais si nous changeons la première ligne en :

```
index: 2
```

L'interférence apparaît. Nous pouvons nous demander si le raffinement /NAME pourrait aider... mais la réponse est négative, ainsi que Romano l'a mentionné (NdT : voir le ticket dans RAMBO).

6 Solution

J'offre la fonction TFUNC afin de résoudre le problème du THROW (ainsi que d'autres problèmes) pour les fonctions nidifiées (voir http://www.compkarori.com/vanilla/d...)

L'interférence entre [catch] et CATCH peut être également résolue en utilisant des THROWs locaux, mais certain peuvent trouver cette solution peu pratique.

Une solution plus simple est d'utiliser le raffinement /NAME pour le THROW, mais ceci devrait être étendu afin de nous permettre de spécifier le nom approprié pour le [catch] également.

Dans ce cas, je pense qu'il serait opportun que la comparaison de noms emploie le = ? à la place de =, ce qui exploiterait la pleine puissance de REBOL.

Fin

- Ladislav