The REBOL Documentation Project

-- FR - Documentation REBOL - Divers --

Divers

Make Function ! doit changer

Philippe Le Goff

Première publication : 28 avril 2006, et mis en ligne le vendredi 28 avril 2006

Résumé :

La traduction du post 3 sur Rebol 3.0, et le changement que devrait connaitre make function !

Carl Sassenrath, CTO REBOL Technologies 8-Apr-2006 22:39 GMT Article #0003

Il existe une contradiction, depuis longtemps, dans l'implémentation de la fonction **make** sous REBOL, qui sera modifiée dans REBOL 3.0. (Mais, cela n'aura que peu d'effet sur la plupart des programmes.)

Si j'interroge REBOL à propos de la fonction **make**, il répond quelque chose comme cela :

```
>> ? make
USAGE:
  MAKE type spec

DESCRIPTION:
  Constructs and returns a new value.
  MAKE is an action value.

ARGUMENTS:
  type -- The datatype or example value. (Type: any-type)
  spec -- The attributes of the new value. (Type: any-type)
```

Donc, si ceci est exact, comment la ligne ci-dessous peut elle être valide ?

```
f: make function! [a] [print a]
```

Dans cet exemple, **make** prend *trois* arguments, mais la description de la fonction mentionne uniquement deux arguments : *type* et *spec*.

Ceci va être changé dans REBOL 3.0. Voici pourquoi :

1. Un programme doit pouvoir s'appuyer avec précision sur les caractéristiques d'interface de fonction. C'est important non seulement pour des choses comme des outils de déboguage, mais également pour la réflexion appropriée (du code vers des données qui renvoient du code).

2. L'ancienne approche nécessitait un hack dans l'implémentation de l'évaluation de REBOL pour "faire marcher" l'argument supplémentaire. Effacer cela simplifierait le code et augmenterait la rapidité de REBOL.

La nouvelle forme de la fonction make sera :

```
f: make function! [[a] [print a]]
```

Cela sera cohérent avec la spécification de l'interface de la fonction.

Comment ceci va-t-il affecter la compatibilité ? Très peu. La plupart des programmes n'utilisent pas cette forme de création de fonction ; ils utilisent les fonctions comme **func**, **function**, **does**, ou **has**.

(Traduction : Philippe Le Goff)