# RFC 9741 : Concise Data Definition Language (CDDL): Additional Control Operators for the Conversion and Processing of Text

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 5 mars 2025

Date de publication du RFC : Mars 2025

https://www.bortzmeyer.org/9741.html

---

Le langage CDDL, qui permet de créer un schéma formel pour des formats comme CBOR, peut s'étendre via l'ajout d'« opérateurs de contrôle ». Ce RFC en spécifie quelques uns, notamment pour agir sur du texte ou convertir d'une forme dans une autre.

CDDL est normalisé dans le RFC 8610 [1]. Sa principale utilisation est pour fournir des schémas afin de valider des données encodées en CBOR (RFC 8949). Du fait de l'extensibilité de CDDL (voir notamment RFC 8610, section 3.8), le RFC 9165 a pu ajouter l'addition, la concaténation, etc. Notre nouveau RFC 9741 ajoute encore d'autres opérateurs, surtout de conversion d'une représentation dans une autre.

Je ne vais pas tous les citer ici (lisez le RFC), juste donner quelques exemples. D'abord, `.b64u`, qui va convertir une chaine d'octets en sa représentation en Base64 (RFC 4648 et, rassurez-vous, il y a aussi des opérateurs pour Base32 et les autres).

Voici un exemple. Mais d'abord, on installe l'outil cddl. Écrit en Ruby, il s'installe avec :

```
% gem install cddl
```

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc8610.txt

Ensuite, on écrit un schéma CDDL utilisant l'opérateur `.b45` (Base45 est normalisé dans le RFC 9285) :

```
%    cat base.cddl
encoded = text .b45 "Café ou thé ?"
```

En utilisant les fonctions de génération de l'outil cddl, on trouve :

```
% cddl base.cddl generate
"EN8R:C6HL34ES44:AD6HLI1"
```

C'est bien l'encodage en Base45 de la chaine de caractères utilisée (évidemment, dans un vrai schéma, on utiliserait une variable, pas une chaine fixe).

Ensuite, `.printf` va formater du texte, en utilisant les descriptions de format du `printf` de C.

```
% cat printf.cddl
my_alg_19 = hexlabel<19>
hexlabel<K> = text .printf (["0x%04x", K])

% cddl printf.cddl generate
"0x0013"
```

Bien sûr, ce n'est pas la méthode la plus simple pour convertir le 19 décimal en 13 hexadécimal. Mais le but est simplement d'illustrer le fonctionnement des nouveaux opérateurs.

Enfin, `.join` va transformer un tableau en chaine d'octets. Voici l'exemple du RFC, avec des adresses IPv4 :

```
% cat address.cddl

legacy-ip-address = text .join legacy-ip-address-elements
legacy-ip-address-elements = [bytetext, ".", bytetext, ".",
                              bytetext, ".", bytetext]
bytetext = text .base10 byte
byte = 0..255

% cddl address.cddl generate
"108.58.234.211"
% cddl address.cddl generate
"42.65.53.156"
% cddl address.cddl generate
"77.233.49.115"
```