Projet de “Programmation Fonctionnelle” 2003

Robert Strandh et David Sherman

2003-02-02

1 Base de donnée de livres

Un document XML est souvent trivial à traduire en notation Lisp (S-expressions). Pour ce projet, nous allons utiliser un tel document correspondant à une base de donnée contenant des descriptions de livres.

Voici une description de la base, une fois lue par `read` :

Un *objet* est soit une *entrée* ou un *attribut* d'une entrée. Il n'y a pas de distinction syntaxique entre une entrée et un attribut, mais un attribut est toujours associé à une entrée, alors qu'une entrée est une valeur autonome.

Une *objet* est une liste dont le premier élément est le *nom* de l'objet et les autres éléments sont des *valeurs de l'objet*. Le nom détermine (par convention) si l'objet est une entrée ou un attribut. Un objet peut être *atomique* ou *non atomique*.

Un objet atomique contient exactement une valeur qui est toujours une valeur atomique comme une chaîne de caractères, un symbole ou un nombre. Un objet atomique est souvent un attribut et peut être utilisé pour représenter des informations simples, par exemple :

- `(price 96.90)` pour un prix,
- `(street "Cours de la Libération")` pour une rue,
- `(phone "+33 556 84 6000")` pour un numéro de téléphone.

Les valeurs d'un objet non atomique sont toujours des listes. Un objet non atomique est soit *singulier* soit un *conteneur*. Un objet singulier contient un nombre arbitraire de valeurs qui sont elles-mêmes des attributs. Voici quelques exemples :

- `(name (first "Bertrand") (last "Short"))`
- `(book (author ...)  (author ...)  (publisher ...)  (title ...))`
- `(address (street ...)  (number ...)  (zip ...))`

Comme vous le voyez dans l'exemple du livre ci-dessus, il peut y avoir plusieurs attributs avec le même nom dans un objet.

Une entrée est le plus souvent un objet non atomique (et dans ce cas singulier) (comme `book`, par exemple), mais un objet non atomique singulier n'est pas forcement une entrée (comme `address`, par exemple).

Dans un conteneur les valeurs représentent une suite ou un ensemble d'éléments. Ainsi, toutes les valeurs sont des objets du même type, par exemple :

- `(reviews (review ...)  (review ...))`
- `(books (book ...)  (book ...)  (book ...))`

Votre base de donnée est un objet non atomique dont le nom est `db` ayant les attributs suivants :

attribut	nom	obliga-toire	multiples possibles	description
auteurs	`authors`	oui	non	un conteneur ; les éléments sont des auteurs
maisons d'éditition	`publishers`	oui	non	un conteneur ; les éléments sont des maisons d'édition
livres	`books`	oui	non	un conteneur ; les éléments sont des livres
comptes rendus	`reviews`	oui	non	un conteneur ; les éléments sont des comptes rendus
sources	`sources`	oui	non	un conteneur ; les éléments sont des sources de vente
prix	`prices`	oui	non	un conteneur ; les éléments sont des prix

Un objet de type *auteur* est objet non atomique contenant les attributs suivants :

attribut	nom	obliga-toire	multiples possibles	description
nom	`last`	oui	non	atomique ; valeur est une chaîne de caractères
prénom	`first`	oui	non	atomique ; valeur est une chaîne de caractères

Un objet de type *maison d'édition* est objet non atomique contenant les attributs suivants :

attribut	nom	obligatoire	multiples possibles	description
nom	`name`	oui	non	atomique ; valeur est une chaîne de caractères

Un objet de type *livre* est un objet non atomique contenant les attributs suivants :

attribut	nom	obligatoire	multiples possibles	description
année	`year`	oui	non	atomique ; valeur est un entier
titre	`title`	oui	non	atomique ; valeur est une chaîne de caractères
auteur	`author`	non	oui	non atomique ; un auteur
éditeur	`editor`	non	oui	non atomique ; un éditeur
maison d'édition	`publisher`	oui	non	non atomique ; une maison d'édition
prix	`price`	non	non	atomique ; valeur est un nombre flottant

Un objet de type *compte rendu* est un objet non atomique contenant les attributs soivants :

attribut	nom	obligatoire	multiples possibles	description
livre	`book`	oui	non	non atomique ; un livre
texte	`text`	oui	non	atomique ; valeur est une chaîne de caractères

Un objet de type *source de vente* est un objet non atomique contenant les attributs suivants :

attribut	nom	obligatoire	multiples possibles	description
nom	`name`	oui	non	atomique ; valeur est une chaîne de caractères

Un objet de type *prix* est un objet non atomique contenant les attributs suivants :

attribut	nom	obliga-toire	multiples possibles	description
livre	`book`	oui	non	non atomique ; un livre
source	`source`	oui	non	non atomique ; une source
prix	`euros`	oui	non	atomique ; valeur est un nombre flottant

Une différence essentielle entre la version XML et la version Lisp de cette base, est que la version XML est un arbre, alors que la version Lisp ne l'ai pas (c'est un graphe sans cycles). Ceci nous permet d'éviter les duplications d'informations. Ainsi, dans la base que vous allez utiliser, chaque objet est unique. Ceci a pour conséquence que vous pouvez tester l'égalité entre deux objets avec la fonction Common Lisp `eq`, sans avoir besoin de comparer les différents attributs.

2 Exemple de base de donnée

Vous trouvez un exemple de base de donnée sur la page d'accueil de cette UV.

3 Travail à faire

Le but du projet est d'utiliser les techniques de la programmation fonctionnelle pour écrire un ensemble d'outils pour répondre à des requêtes dans une base de donnée de livres en particulier, mais aussi plus généralement dans une base de donnée structurée selon les règles indiquées ci-dessus.

Vous devez tester votre logiciel sur les requêtes suivantes (le résultat doit être une liste contenant les informations indiquées) :

1. Tous les livres de Addision-Wesley publiés après 1991, où chaque livre est indiqué par son titre et son année de publication.
2. Tous les couples contenant l'auteur et le titre du livre, triée par auteur.
3. Tous les livres disponibles à la fois sur Amazon.com et BN.com, avec le prix chez les deux fournisseurs.
4. Tous les livres dont le titre mentionne le mot “Lisp”.
5. Tous les livres dans la base publié après 1993 et dont le titre contient le mot “programming” (sans tenir compte de la différence entre majuscules et minuscules).

6. Les noms des maisons d'édition ayant au moins un livre de Paul Graham ou de Peter Norvig.

7. Toutes les maisons d'édition dans la base n'ayant aucun livre dans la base.

8. Toutes les maisons d'édition ayant au moins deux livres dans la base.

Soyez sûrs de respecter les règles de l'espacement et de l'indentation habituellement utilisés pour Common Lisp. Il est important de commencer à s'habituer à respecter la culture partagée par les membres d'une communauté de programmeurs. Pour cela n'hésitez pas regarder le code disponible en ligne des auteurs de livres sur Lisp cités dans votre base de données de test. Un projet qui ne respecte pas ces règles verra sa note diminuée de 2 à 4 points (sur 20) selon la gravité de la situation. Soyez sûrs également d'utiliser la bonne bibliothèque d'indentation sous Emacs. Par défaut, c'est la bibliothèque pour Emacs Lisp qui est utilisée. Il faut utiliser celle destinée à l'indentation de Common Lisp.