Mar 5, 2019

1. Objectifs du TP

- Utiliser des Modules Ruby.
- Apprendre à documenter son code.

2. Petits sujets variés

2.1. Utilisation de module préféfini: un arbre binaire

Écrire conteneur type *arbre binaire trié*. À l'insertion dans un sous-arbre, on compare l'élément à la racine: si celui-ci est plus petit, on insère à gauche, sinon on insère à droite.

- À quoi va ressembler un noeud de l'arbre?
- Comment bénéficier de toutes les méthodes pour les conteneurs?
- Quelles méthode doit donc implémenter l'arbre? Comment l'écrire (algo, pas ruby…).
- Comment être sûr de gagner du temps sur les recherches dans l'arbre (rappel, il est *trié*)?
- Testez.

2.2. Écriture d'un module de mixin

- Écrivez un module `Unaccent` qui permet d'ajouter à toute classe possédant une méthode `to_s` une méthode `to_us` où tous les éventuels accents ont été simplifiés.
- Ce module doit exporter des conversions. Pour ne pas perdre de temps, je vous la donne: `[[ 'êéèë','e' ], ['ùûü','u'], ['ç','c'], ['ôö','o'], ['àâä','a'], ['îï','i']]`. Si vous voulez les

majuscules, vous savez ce qu'il vous reste à faire!
- N'oubliez pas de mettre au début de votre fichier `#encoding: utf-8` pour éviter les problèmes sur toutes les versions de Ruby.
- Testez votre module avec une classe de votre facture.
- Et maintenant, comment faire pour que la classe `String` possède aussi un `to_us`?

2.3. Documentation d'une class ruby

Comment ça, personne ne m'a encore demandé comment documenter son code en ligne? Bon, en bref, on a deux solutions:

- Rdoc est la solution la plus ancienne et la plus classique. La doc officielle devrait vous suffire. La syntaxe tient beaucoup du Markdown (on en profite pour se renseigner sur Aaron Swartz, un jeune au cursus impressionnant…) mais la gestions des arguments et retours n'est pas du goût de tout le monde (en gros, on en fait juste des sections Markdown).
- Yard est plus récent, et sa syntaxe reprend le standard de fait à la doxygen (`@param`, `@return`…), avec en plus des ajouts pour préciser des types masqué dans les prototypes par le duck-typing. La mise en forme en Markdown est possible. De plus, il est compatible avec *rdoc*. Je suis donc tenté de vous conseiller plutôt cette solution. Vous pouvez commencer ici.
- Voilà, je vous laisse expérimer sur votre code.

3. Pour aller plus loin

- Vous trouverez ici un bon article qui revient sur les différences entre bloc, lambda, et proc.

Supports de cours pour l'IUT
francois.delobel@uca.fr

TPs mis à disposition le plus simplement possible pour mes étudiants.