

Precedences

Gestion de précédences dans les listes de choix pour `Tty-Prompt#select`.

Deux modes d'utilisation sont possibles.

Le plus simple, par block :

```
require 'precedences'

#
# Tty-prompt#select choices
#
choices = [
  {name:"Choix premier", value: :first},
  {name:"Deuxième choix", value: :second}
]

#
# Fichier où les précédences seront enregistrées
#
precfile = File.join(__dir__, '.precedences')

choix = precedencize(choices, precfile) do |q|
  q.question "Choisir la valeur :"
end
```

`choix` contiendra la valeur choisie, soit `:first` soit `:second`.

Le second mode d'utilisation fonctionne sans bloc, avec les valeurs par défaut :

```
require 'precedences'

##
# On définit choices et prefile de la même façon
# puis...

choices = precedencize(choices, precfile)

choix = Q.select("Choisir parmi : ", choices)
#
# Pour enregistrer cette précédence (ne pas l'oublier !)
#
set_precedence(choix)

# ...
```

Valeurs possibles

Dans l'utilisation normale, l'attribut `:value` des choix doit obligatoirement être de type `String`, `Symbol` ou `Numeric`, mais avec l'option `precedences_per_index`, il est possible d'utiliser n'importe quelle valeur (note : l'ordre est alors mémorisé par index — ce qui signifie qu'il ne faut pas modifier la liste en cours de route).

Par exemple :

```
require 'precedences'

#
# Des choix avec des valeurs spéciales
#
choices = [
  {name:"La classe Integer" , value: Integer},
  {name:"La classe Array"   , value: Array},
  {name:"La classe Hash"    , value: Hash},
]

choix = precedencize(choices, file) do |q|
  q.question "Choisis une classe"
  q.precedences_per_index # <=== option pour que ça passe
end
```

Options possibles

En mode block, on peut définir plusieurs choses :

```
require 'precedences'

choix = precedencize(choices, precfile) do |q|
  #
  # La question
  #
  q.question = "Ma question"
  # ou
  q.question "Ma question"

  #
  # Le nombre de menus affichés
  # (noter que par défaut, tous les menus sont affichés, contrairement
  # à tty-prompt qui les limite toujours)
  #
  q.per_page 5
  # ou q.per_page = 5
end
```

```

#
# L'affichage ou non de l'aide (:never par défaut)
#
q.show_help = :always
# ou
q.show_help :start

#
# Le message d'aide à afficher
#
q.help = "Mon message d'aide"
# ou
q.help "Message d'aide"

#
# La valeur sélectionnée par défaut (lorsqu'il faut passer
# outre l'ordre de précedence par exemple)
# Cette valeur peut être passée par...
#
# Par index 1-based explicite
q.default = 4
# ou q.default 4
#
# Par valeur du :value
q.default = :second
# ou q.default :second
#
# Par extrait du name (ou name exact)
q.default = "premier"
# ou q.default "premier"
# => Sélectionnera le choix "Choix premier"

end

```

Ajouter un menu “Renoncer”

```

choix = precedencize(choices, file) do |q|
  q.add_choice_cancel(:up, {value: :cancel, name: "Renoncer"})
end

```

Si on doit utiliser les valeurs par défaut que sont :

- `:name` est “Cancel”
- `:value` est `nil`
- position est `:down` alors on peut faire simplement

```
choix = precedencize(choices, file) do |q|  
  q.add_choice_cancel  
end
```



Noter qu'on peut en fait se servir de ce menu pour ajouter n'importe quel autre menu que "Renoncer".