

LE LANGAGE RUBY

Sommaire

1. Presentation generale
2. Installation et utilisation
3. Exemple de code
4. Le framework Ruby on Rails

1. Presentation generale

Ruby est un langage de programmation libre apparu au Japon en 1995 et standardisé depuis 2011. Son créateur, Yukihiro Matsumoto, l'a conçu comme un ensemble homogène intégrant un best-of des fonctionnalités de ses langages préférés de l'époque, notamment Perl, Smalltalk, Eiffel, Ada et Lisp.



Yukihiro MATSUMOTO en pleine réflexion

Souvent décrit comme un équivalent à Python, Ruby est un langage multi-paradigme qui mêle astucieusement programmations impérative et fonctionnelle à une philosophie Objet.

Ruby n'est pas uniquement destiné à faire du web. Il est également utilisé pour automatiser des tâches récurrentes ou pour développer des applications, de la bureautique aux applications mobiles en passant par les jeux.

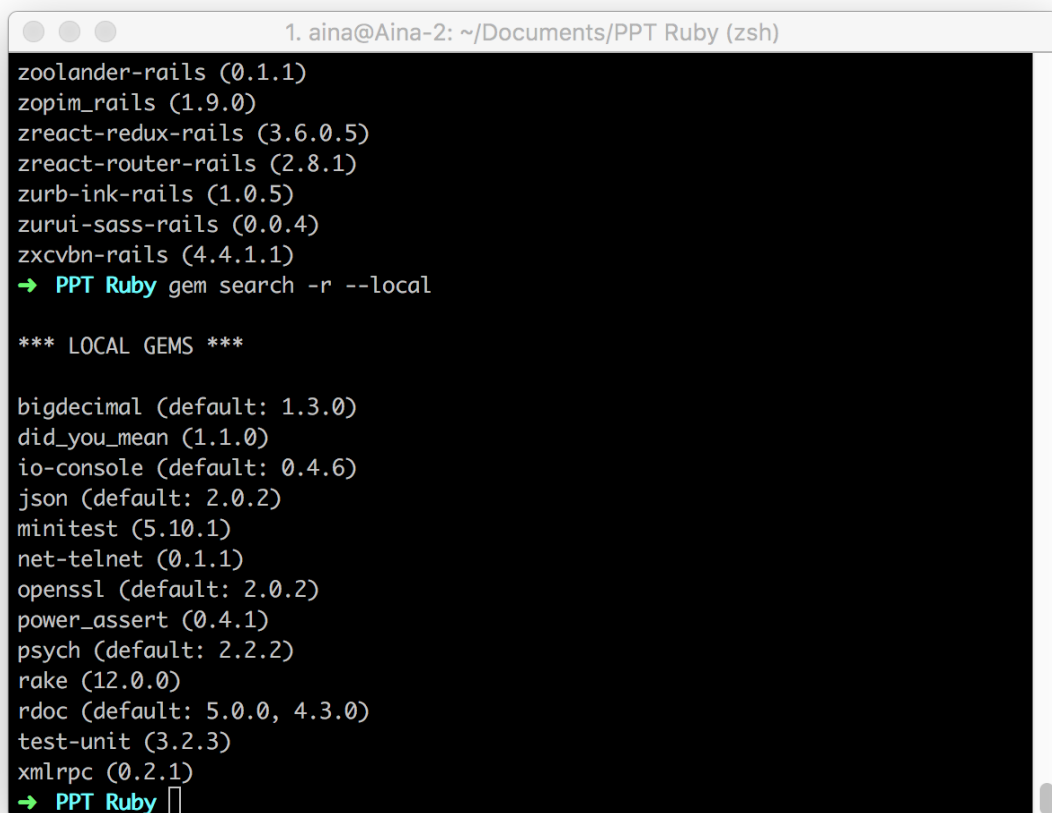
Ce langage s'appuie sur un écosystème de bibliothèques, baptisées gems. Ces composants permettent d'augmenter les capacités du langage pour répondre aux besoins des développeurs. RubyGems, "le gestionnaire de paquets de Ruby", permet de les télécharger et de les installer. Il facilite également leur création et leur partage.

Qu'est-ce qu'un gem ?

Comme la plupart des langages de programmation actuels, Ruby s'appuie sur tout un écosystème de bibliothèques logicielles (ou "librairies", de l'anglais "library" : bibliothèque).

Ces bibliothèques sont bien souvent disponibles sous la forme d'une *gem*. RubyGems est le « gestionnaire de paquets de Ruby » permettant de télécharger et d'installer ces gems. Il facilite également leur création et leur partage. Il s'agit donc d'un équivalent à Apt ou Homebrew (gestionnaires de paquets Debian ou Mac respectivement), mais pour Ruby spécifiquement.

Depuis Ruby 1.9, RubyGems est intégré dans Ruby. Pour les versions précédentes, il faudra l'installer séparément.

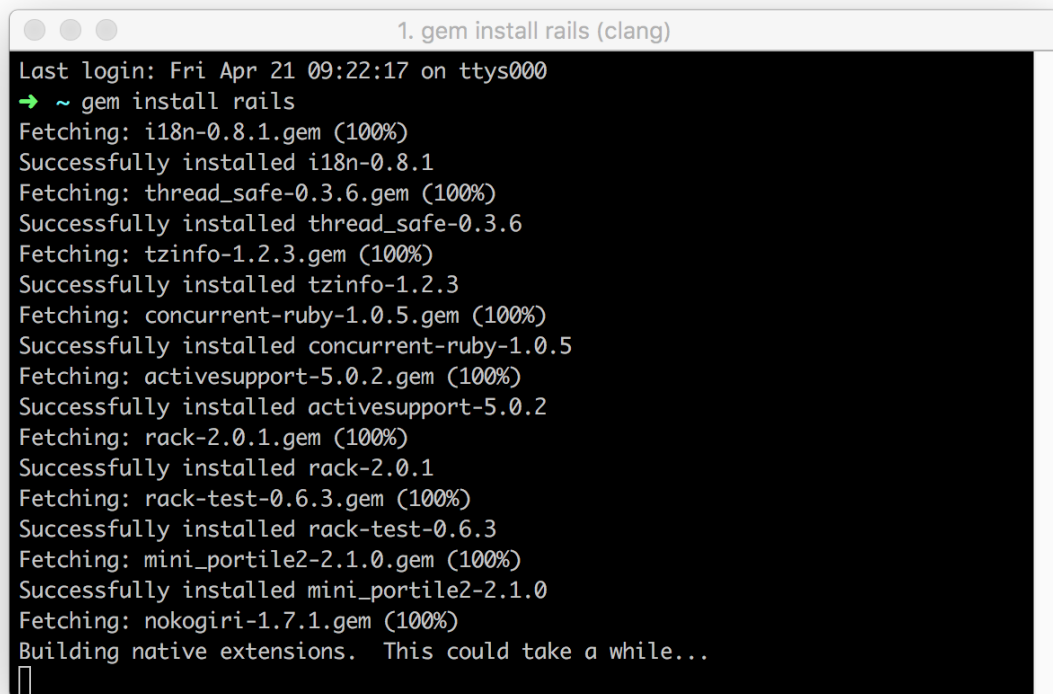


```
1. aina@Aina-2: ~/Documents/PPT Ruby (zsh)
zoolander-rails (0.1.1)
zopim_rails (1.9.0)
zreact-redux-rails (3.6.0.5)
zreact-router-rails (2.8.1)
zurb-ink-rails (1.0.5)
zurui-sass-rails (0.0.4)
zxcvbn-rails (4.4.1.1)
→ PPT Ruby gem search -r --local

*** LOCAL GEMS ***

bigdecimal (default: 1.3.0)
did_you_mean (1.1.0)
io-console (default: 0.4.6)
json (default: 2.0.2)
minitest (5.10.1)
net-telnet (0.1.1)
openssl (default: 2.0.2)
power_assert (0.4.1)
psych (default: 2.2.2)
rake (12.0.0)
rdoc (default: 5.0.0, 4.3.0)
test-unit (3.2.3)
xmlrpc (0.2.1)
→ PPT Ruby
```

Ici on utilise la commande **gem search -r --local** pour trouver la liste des gems installés localement

A terminal window titled "1. gem install rails (clang)" with a dark background and light text. It shows the output of the command "gem install rails". The output lists the fetching and successful installation of several gems: i18n-0.8.1, thread_safe-0.3.6, tzinfo-1.2.3, concurrent-ruby-1.0.5, activesupport-5.0.2, rack-2.0.1, rack-test-0.6.3, mini_portile2-2.1.0, and nokogiri-1.7.1. It ends with the message "Building native extensions. This could take a while..." and a cursor.

```
1. gem install rails (clang)
Last login: Fri Apr 21 09:22:17 on ttys000
➔ ~ gem install rails
Fetching: i18n-0.8.1.gem (100%)
Successfully installed i18n-0.8.1
Fetching: thread_safe-0.3.6.gem (100%)
Successfully installed thread_safe-0.3.6
Fetching: tzinfo-1.2.3.gem (100%)
Successfully installed tzinfo-1.2.3
Fetching: concurrent-ruby-1.0.5.gem (100%)
Successfully installed concurrent-ruby-1.0.5
Fetching: activesupport-5.0.2.gem (100%)
Successfully installed activesupport-5.0.2
Fetching: rack-2.0.1.gem (100%)
Successfully installed rack-2.0.1
Fetching: rack-test-0.6.3.gem (100%)
Successfully installed rack-test-0.6.3
Fetching: mini_portile2-2.1.0.gem (100%)
Successfully installed mini_portile2-2.1.0
Fetching: nokogiri-1.7.1.gem (100%)
Building native extensions. This could take a while...
█
```

*La commande `*gem install rails*` nous permet d'installer le framework Ruby on Rails contenant une multitude de gem*

2. Installation et Utilisation

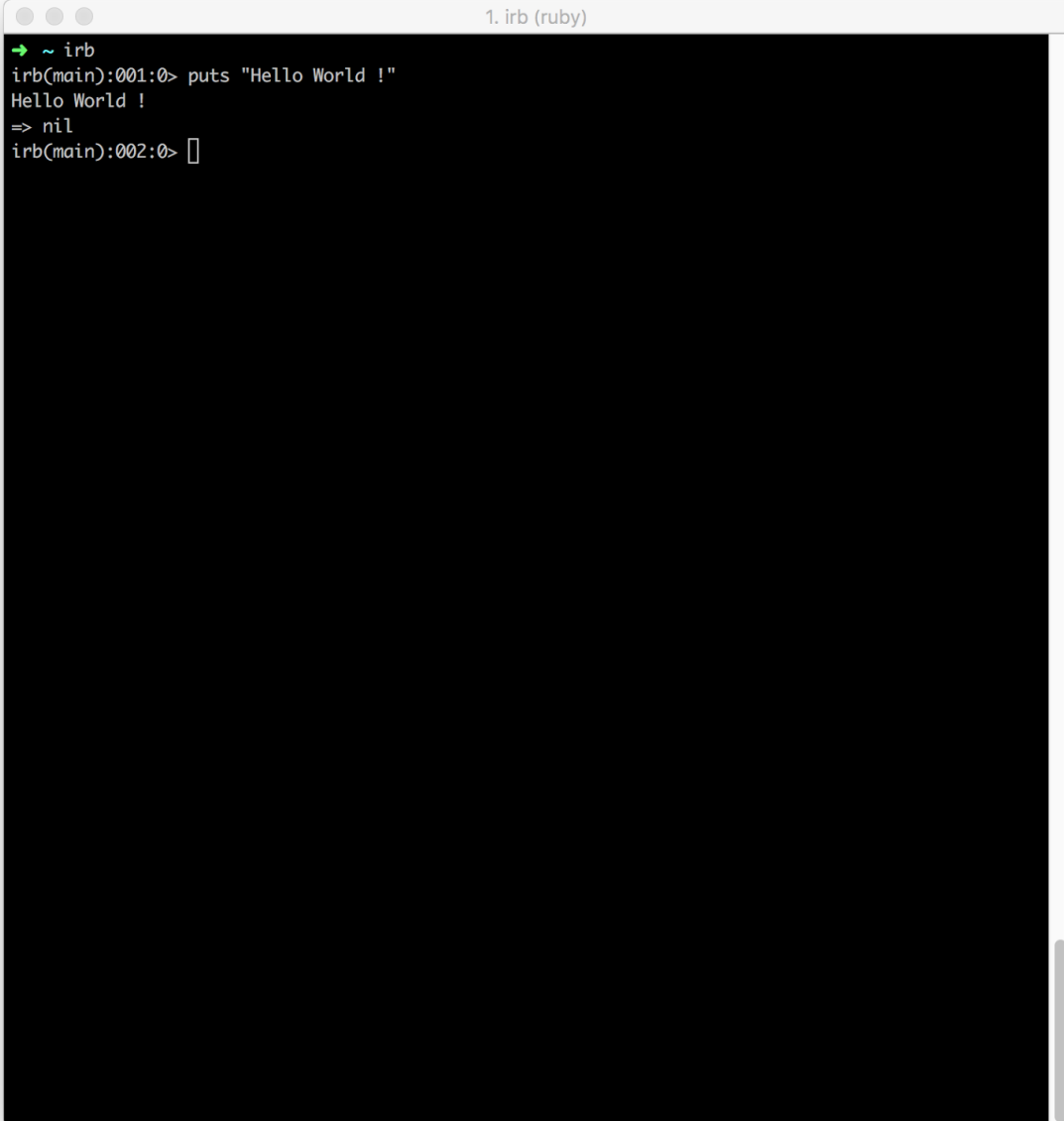
Il existe plusieurs manières d'installer Ruby :

- Si vous êtes sur un système d'exploitation de type UNIX, utiliser le **gestionnaire de paquets** de votre système est la façon la plus simple de procéder. Toutefois, la dernière version de Ruby pourrait ne pas être disponible.
- Un **Installateurs** peut être utilisé pour installer une ou plusieurs versions de Ruby. Il existe aussi un installateur pour Windows.
- Des **Managers** aident à basculer d'une version de Ruby à une autre sur votre système.
- Et finalement, vous pouvez aussi **compiler Ruby à partir des sources**.

Sur OS X Yosemite et Mavericks, Ruby 2.0 est déjà présent. OS X Mountain Lion, Lion, et Snow Leopard sont fournis avec Ruby 1.8.7.

Pour les utilisateurs de la distribution Debian, la commande `*sudo apt-get install ruby-full*` suffit pour récupérer la dernière version de Ruby disponible

Pour commencer à utiliser Ruby, ouvrez votre terminal et tapez la commande `*irb*` l'interpréteur se lancera :

A screenshot of a terminal window with a light gray title bar containing three window control buttons and the text "1. irb (ruby)". The terminal has a black background with white text. It shows the command prompt "~ irb" with a green arrow icon. Below it, the prompt "irb(main):001:0>" is followed by the command "puts 'Hello World !'", which outputs "Hello World !". The next line shows the prompt "irb(main):002:0>" followed by "=> nil".

```
1. irb (ruby)
➔ ~ irb
irb(main):001:0> puts "Hello World !"
Hello World !
=> nil
irb(main):002:0> 
```

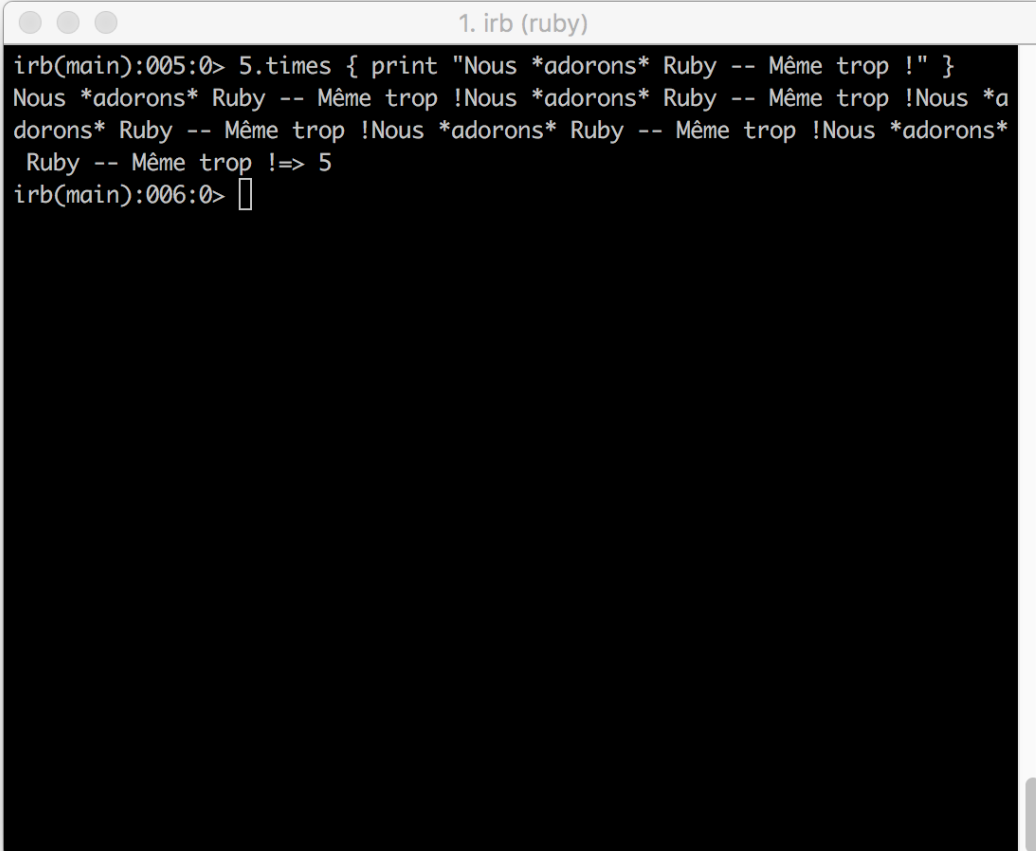
Vous pourrez ainsi commencer à coder en Ruby

3. Exemple de code

Le langage ruby, en terme de syntaxe et de fonctionnalité, a été dès le départ conçu comme un ensemble homogène.

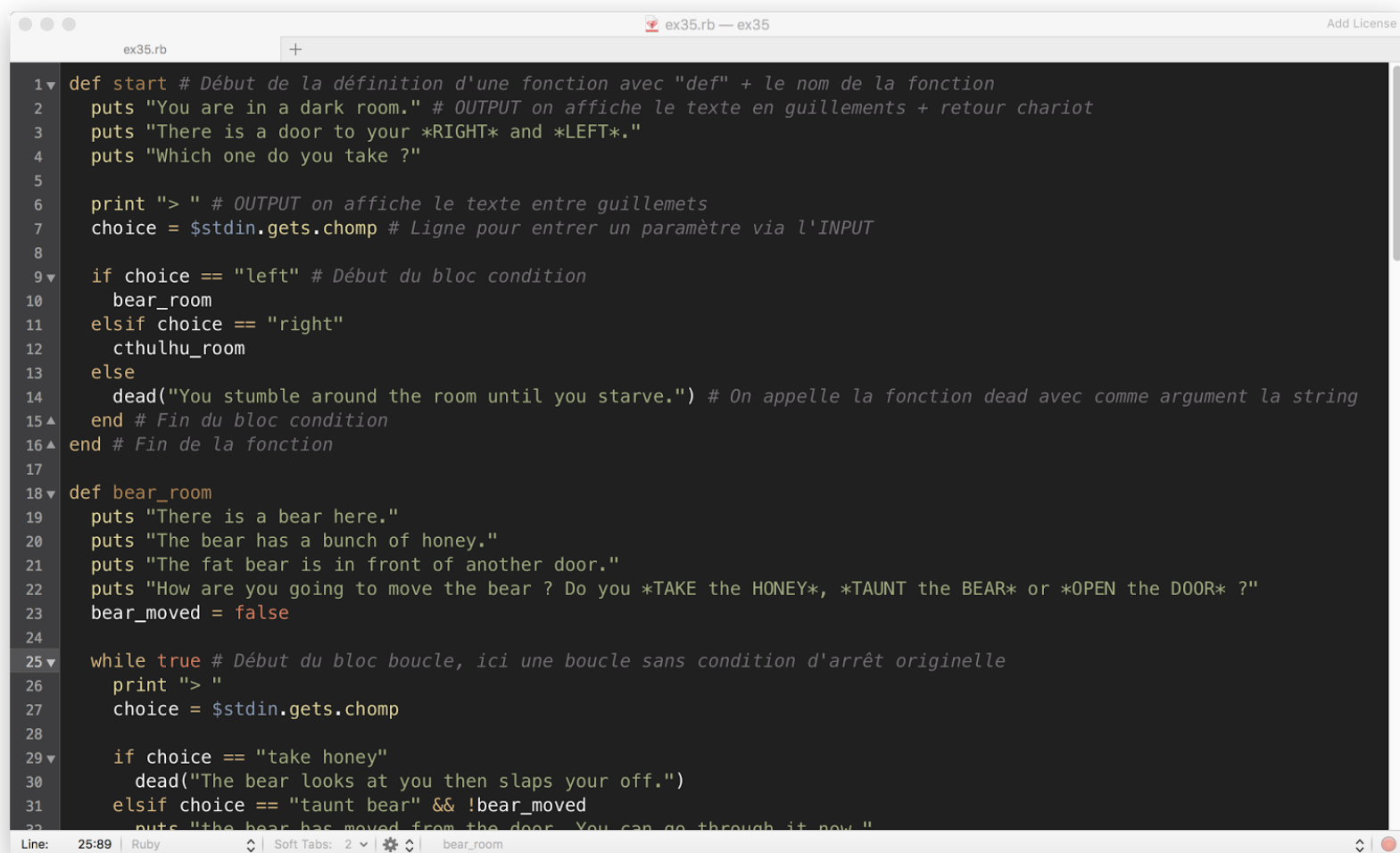
Dans Ruby, le paradigme de base est que tout y est un objet. Chaque entité d'information et de code peut recevoir ses propres propriétés et actions. L'approche purement objet de Ruby est très souvent illustrée par un bout de code montrant l'application d'une action à un nombre.

```
5.times { print "Nous *adorons* Ruby -- Même trop !" }
```



```
1. irb (ruby)
irb(main):005:0> 5.times { print "Nous *adorons* Ruby -- Même trop !" }
Nous *adorons* Ruby -- Même trop !Nous *adorons* Ruby -- Même trop !Nous *a
dorons* Ruby -- Même trop !Nous *adorons* Ruby -- Même trop !Nous *adorons*
Ruby -- Même trop !=> 5
irb(main):006:0> 
```

Au nombre 5 on lui applique la méthode print qui affichera “Nous *adorerons* Ruby -- Même trop !”



```
1 def start # Début de la définition d'une fonction avec "def" + le nom de la fonction
2   puts "You are in a dark room." # OUTPUT on affiche le texte en guillemets + retour chariot
3   puts "There is a door to your *RIGHT* and *LEFT*."
4   puts "Which one do you take ?"
5
6   print "> " # OUTPUT on affiche le texte entre guillemets
7   choice = $stdin.gets.chomp # Ligne pour entrer un paramètre via l'INPUT
8
9   if choice == "left" # Début du bloc condition
10    bear_room
11  elsif choice == "right"
12    cthulhu_room
13  else
14    dead("You stumble around the room until you starve.") # On appelle la fonction dead avec comme argument la string
15  end # Fin du bloc condition
16 end # Fin de la fonction
17
18 def bear_room
19   puts "There is a bear here."
20   puts "The bear has a bunch of honey."
21   puts "The fat bear is in front of another door."
22   puts "How are you going to move the bear ? Do you *TAKE the HONEY*, *TAUNT the BEAR* or *OPEN the DOOR* ?"
23   bear_moved = false
24
25   while true # Début du bloc boucle, ici une boucle sans condition d'arrêt originelle
26     print "> "
27     choice = $stdin.gets.chomp
28
29     if choice == "take honey"
30       dead("The bear looks at you then slaps your off.")
31     elsif choice == "taunt bear" && !bear_moved
32       puts "The bear has moved from the door. You can go through it now "
```

Exemple de code Ruby avec des fonctions, des conditions et des boucles

4. Ruby on rails

La première version de Ruby on Rails date de juillet 2004. Le framework a été extrait de Basecamp, un outil de gestion de projets développé par David Heinemeier Hansson.

La première version stable (1.0) est sortie le 14 décembre 2005.

À partir de cette date, Ruby on Rails a marqué le monde du développement web. On a vu pousser un ensemble de frameworks web *Rails-like* (notamment CakePHP, Symfony et CodeIgniter dans le monde PHP).

Une *preview release* de la version 2.0 a été annoncée le 30 septembre 2007. Rails 2.0 apporte principalement :

- les *resources* qui fournissent une architecture REST ;
- une différenciation plus claire entre les formats et les convertisseurs (un fichier .rhtml devient un fichier .html.erb, c'est-à-dire un fichier interprété par eruby et dont le résultat est de l'HTML) ;
- la possibilité d'utiliser des modèles comme URL (par exemple `redirect_to(person)`) ;
- la gestion d'authentification HTTP basique ;
- une protection contre les attaques CSRF ;
- l'interception plus simple d'exceptions génériques ;
- le cache du résultat de certaines requêtes SQL ;
- une simplification de l'écriture de migration ;
- le déplacement dans des plugins de tous les *act_as_** et des modules de base de données propriétaires ;
- le début de l'abandon de SOAP au profit de REST ;
- un nouveau système pour le *déboguer* qui remplace les *breakpoints*.

Rails 2.3 apporte l'usage du middleware Rack qui permet de gérer des requêtes de niveau serveur, et intègre un reverse proxy. Une autre modification importante est la gestion des formulaires enfants d'un objet parent (*nested form*).

Rails 3 est la rencontre entre Rails et Merb. À la suite d'une réécriture complète, agnostique (l'API générale est déconnectée d'une bibliothèque particulière). La version 3.0 est sortie le 29 août 2010.

La version 4 est sortie le 25 juin 2013 et conseille l'usage de Ruby 2.

Ruby on Rails est un Framework open source utilisant le langage Ruby pour développer des applications web. Associé à une méthodologie Agile, il permet le développement très rapide d'application web. L'avantage majeur de Ruby on Rails est donc sa productivité élevée. Ruby on Rails répond pleinement aux problématiques complexes, aux demandes particulières et réduit fortement le temps de développement des applications web, tout en maintenant une très forte qualité. Ruby on rails accompagné des méthodes de gestion de projet Agile, est l'assurance d'avoir un site web innovant, évolutif et performant. Pour aller un peu plus loin, commençons par une présentation du langage Ruby, le langage sur lequel repose Ruby on Rails.

Un framework complet

Ruby on Rails est conçu sur le modèle MVC (Modèle : ActiveRecord – Vue : ActionView – Contrôleur : ActionController). Plus que tout autre Framework, Ruby on Rails est associé à une philosophie : le pragmatisme illustré par « Convention over configuration ». Ce qui en fait un outil interactif et d'une certaine simplicité. Les conventions permettent

d'éviter de penser à la configuration avant de procéder au développement. La configuration se fera au fur et à mesure du besoin du développement. Cela n'implique pas que l'on ne peut pas configurer, au contraire on peut même configurer tout ce que l'on veut. La programmation par convention n'élimine pas simplement le besoin de configuration immédiat mais il réduit aussi les lignes de codes. Le code est structuré et dispose d'une infrastructure puissante. C'est une opposition radicale à d'autres Framework et c'est ce qui en fait l'un des langages le plus « beau » et lisible. Un besoin qui, vous l'aurez compris, se fait vite ressentir pour les applications complexes, tel qu'une application métier ou pour la reprise du code lors de l'évolution de l'application.

C'est un Framework qui encourage aux bonnes pratiques par les outils de tests qui sont à disposition et par l'automatisation, c'est à dire les générateurs de codes fournis avec RoR. Les générateurs permettent de créer le code qui est souvent utilisé, permettant de ne pas réinventer la roue à chaque fois. Utiliser ces générateurs est un gain de temps énorme et ils permettent de réaliser des sites web encore plus rapidement.

Véritable outil à développer des applications web riches de fonctionnalités et d'interactivités, ses nombreuses qualités nous ont

conduit à nous spécialiser dans ce Framework pour le développement de sites web innovants et dynamiques.