4

Les entrées/sorties

Dans ce chapitre, vous allez voir comment Scilab s'intègre avec son environnement informatique. De plus, vous trouverez ici des exemples simples de lignes de commandes exécutées dans la console de Scilab, qui vous permettront de vous habituer à l'utilisation de la console.

4.1. Système de fichiers

La première notion à comprendre pour bien travailler avec Scilab est celle de répertoire courant. À chaque instant, un répertoire de votre système de fichiers est associé à votre espace de travail dans Scilab. Par défaut, c'est dans ce répertoire que Scilab ira chercher des informations annexes lors de certaines opérations (ouverture ou écriture de fichiers par exemple). Vous pouvez intervenir sur le système de fichier de plusieurs manières :

- récupérer le nom du répertoire courant, par le menu FICHIER de la console ou par la commande `pwd` ;
- changer de répertoire courant, par le menu FICHIER de la console ou en utilisant les commandes `cd` ou `chdir` ;
- créer un nouveau répertoire avec `mkdir` et détruire un répertoire avec `rmdir` ;
- déplacer, copier ou détruire des fichiers directement depuis la console à l'aide des commandes `copyfile`, `movefile` et `mdelete`.

Le chemin de certains répertoires spécifiques à Scilab sont accessibles dans la console à l'aide des commandes suivantes :

- Le répertoire d'installation de Scilab s'obtient avec `SCI` (voir aussi Section 6.2, Installation).
- Le répertoire contenant les données de session d'un utilisateur (historique, préférences, etc.) s'obtient avec `SCIHOME`. Ce répertoire varie selon le système d'exploitation et la manière dont sont gérés les comptes utilisateurs. Par exemple, ce répertoire sera en général :

Cet extrait provient du livre *Scilab : De la théorie à la pratique – I. Les fondamentaux*
écrit par Philippe Roux – © 2013 Éditions D-BookeR

  - `C:/Users/<User>/AppData/Roaming/Scilab/<Scilab-Version>` sur Windows ;
  - `/home/<User>/.Scilab/<Scilab-Version>` sur les systèmes de type Unix ;
  - `/Users/<User>/.Scilab/<Scilab-Version>` sur Mac OS.
- Un répertoire temporaire est aussi associé à chaque session de travail avec Scilab. Il est créé au début de chaque session, puis détruit à la fin. Son chemin s'obtient dans la console avec `TMPDIR`. Son nom est de la forme `SCI_TMP_*` et son emplacement dépend du système d'exploitation.

Voici quelques exemples de manipulations sur les répertoires que vous pouvez effectuer depuis la console :

```
-->path=pwd();  // répertoire courant

-->cd SCI  // aller dans le répertoire d'installation de Scilab
 ans  =

 C:\Program Files\scilab-5.5.0

-->pwd  // valeur du répertoire courant
 ans  =

 C:\Program Files\scilab-5.5.0

-->cd contrib  // aller dans le répertoire SCI/contrib/
 ans  =

 C:\Program Files\scilab-5.5.0\contrib

-->cd '../'  // "remonter" dans le répertoire SCI
 ans  =

 C:\Program Files\scilab-5.5.0

-->chdir('contrib')  // aller dans le répertoire SCI/contrib/
 ans  =

  T

-->pwd  // valeur du répertoire courant
 ans  =

 C:\Program Files\scilab-5.5.0\contrib

-->chdir(TMPDIR)  // aller dans le répertoire temporaire
 ans  =

  T
```

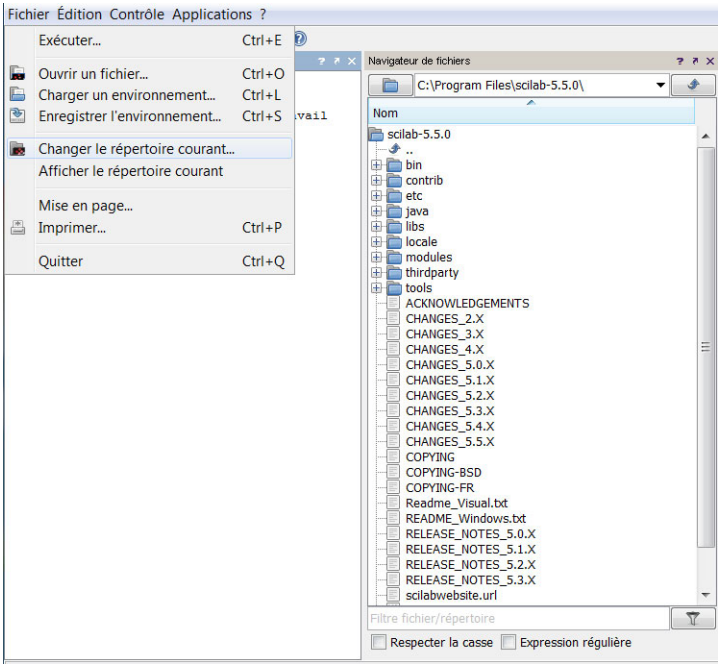
Cet extrait provient du livre Scilab : De la théorie à la pratique – I. Les fondamentaux
écrit par Philippe Roux – © 2013 Éditions D-BookeR

```
-->mkdir('essai')  //créer le répertoire essai/
 ans  =

    1.

-->ls('ess*') //lister les éléments commençant par "ess"
 ans  =

!essai.txt  !
!           !
!essai      !

-->rmdir('essai')  //détruire le répertoire essai/
 ans  =

    1.

-->dir('ess*') //c'est le contenu du répertoire courant qui est vide []
 ans  =

essai.txt

-->chdir(path)  // retour au répertoire courant de départ
 ans  =

  T
```

Toutes ces opérations peuvent également être réalisées *via* une interface graphique à l'aide du navigateur de fichiers (voir Figure 4.1). Ce navigateur peut être appelé depuis le menu APPLICATIONS de la console ou avec la commande `filebrowser`.

Cet extrait provient du livre *Scilab : De la théorie à la pratique – I. Les fondamentaux*
écrit par Philippe Roux – © 2013 Éditions D-BookeR

Figure 4.1 : Le navigateur de fichiers



Astuce *› Quand on lance Scilab depuis un raccourci, on peut paramétrer ce raccourci pour spécifier le répertoire courant de démarrage. Par défaut, ce répertoire est en général le répertoire racine de l'utilisateur qui lance Scilab ou le répertoire d'installation de Scilab (`SCI`) sous Windows.*

4.2. Commandes système

Vous pouvez appeler des commandes système depuis Scilab. Selon le système d'exploitation sur lequel vous travaillez, vous utiliserez plutôt la commande `dos` ou `unix`, mais les deux commandes fonctionnent de la même manière ! Quatre variantes de la commande `unix` traitent différemment le résultat renvoyé par le système :

- `unix_g` : permet de rediriger la sortie vers une variable Scilab ;
- `unix_w` : permet de rediriger la sortie vers la console ;
- `unix_x` : permet de rediriger la sortie vers une fenêtre popup (voir Figure 4.2) ;
- `unix_s` : ne renvoie rien.

Cet extrait provient du livre Scilab : De la théorie à la pratique - I. Les fondamentaux
écrit par Philippe Roux - © 2013 Éditions D-BookeR

Voici quelques exemples de résultats affichés dans la console :

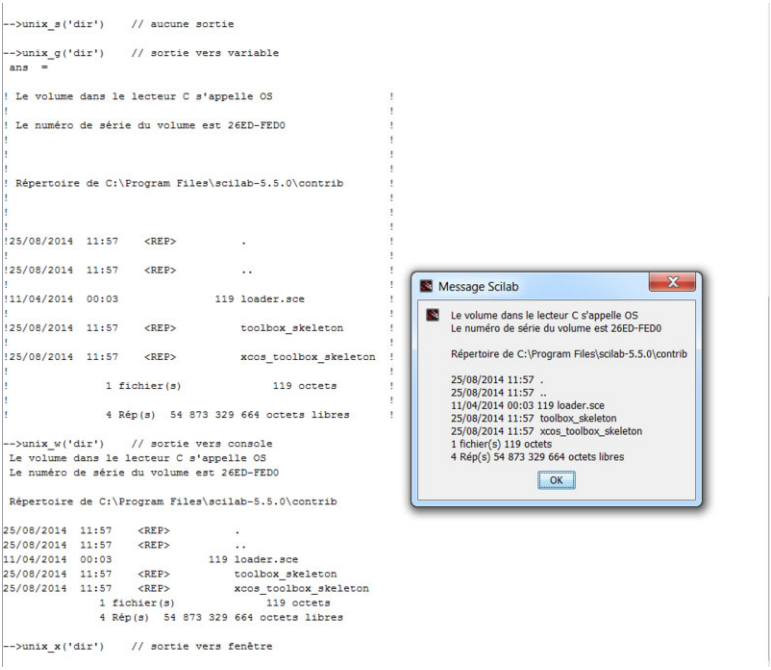
```
-->path=pwd();    // répertoire courant

-->cd SCI    // aller dans le répertoire d'installation de Scilab
 ans  =

 C:\Program Files\scilab-5.5.0

-->cd contrib    // aller dans le répertoire contrib
 ans  =

 C:\Program Files\scilab-5.5.0\contrib

-->unix('dir')      // code retour
 ans  =

    1.

-->unix_s('dir')    // aucune sortie

-->unix_g('dir')    // sortie vers variable
 ans  =

! Le volume dans le lecteur C s'appelle OS                  !
!                                                           !
! Le numéro de série du volume est 26ED-FED0                !
!                                                           !
!                                                           !
!                                                           !
! Répertoire de C:\Program Files\scilab-5.5.0\contrib       !
!                                                           !
!                                                           !
!                                                           !
!25/08/2014  11:57    <REP>          .                      !
!                                                           !
!25/08/2014  11:57    <REP>          ..                     !
!                                                           !
!11/04/2014  00:03               119 loader.sce             !
!                                                           !
!25/08/2014  11:57    <REP>          toolbox_skeleton       !
!                                                           !
!25/08/2014  11:57    <REP>          xcos_toolbox_skeleton  !
!                                                           !
!               1 fichier(s)              119 octets        !
!                                                           !
!               4 Rép(s)  54 873 329 664 octets libres      !

-->unix_w('dir')    // sortie vers console
 Le volume dans le lecteur C s'appelle OS
 Le numéro de série du volume est 26ED-FED0

 Répertoire de C:\Program Files\scilab-5.5.0\contrib

25/08/2014  11:57    <REP>          .
```

Cet extrait provient du livre Scilab : De la théorie à la pratique – I. Les fondamentaux
écrit par Philippe Roux – © 2013 Éditions D-BookeR

```
   25/08/2014  11:57    <REP>          ..
   11/04/2014  00:03                 119 loader.sce
   25/08/2014  11:57    <REP>          toolbox_skeleton
   25/08/2014  11:57    <REP>          xcos_toolbox_skeleton
                  1 fichier(s)              119 octets
                  4 Rép(s)  54 873 329 664 octets libres

   -->unix_x('dir')    // sortie vers fenêtre

   -->cd(path);
```

Attention › *La commande* `unix`*, pour sa part, renvoie un code entier selon le résultat obtenu.*

Figure 4.2 : *Exemple de fenêtre popup renvoyée par* `unix_x`



Pour récupérer les informations concernant le système d'exploitation sur lequel s'exécute Scilab, vous utiliserez la commande `getos`. De même, la commande `getversion` vous permettra de connaître le numéro de la version de Scilab utilisée. Plus généralement, il est possible de récupérer une variable d'environnement du système avec la commande `getenv`.

Cet extrait provient du livre Scilab : De la théorie à la pratique - I. Les fondamentaux
écrit par Philippe Roux - © 2013 Éditions D-BookeR

```
-->getversion()       // version Scilab
 ans  =

 scilab-5.5.0

-->getos()           // os windows
 ans  =

 Windows

-->getenv('TZ')      // récupérer la variable d'environnement TZ
 ans  =

 Europe/Paris
```

En outre, vous pouvez interagir avec le presse-papier en utilisant `clipboard`.

```
-->clipboard("copy","essai")     // CTRL+C sur le texte "essai"
 ans  =

     []

-->clipboard("paste")           // CTRL+V
 ans  =

 essai
```

4.3. Dates et temps CPU

Pour certaines applications, vous pouvez avoir besoin d'accéder à des informations temporelles. Différentes commandes permettent d'évaluer des durées. Vous pouvez ainsi :

- calculer le temps CPU (= nombre de cycle du processeur) écoulé entre deux actions avec `timer` ;
- calculer le temps réel écoulé en millisecondes entre deux actions avec `tic`/`toc`, qui démarre/arrête le chronomètre de Scilab ;
- mettre en pause Scilab pendant un temps donné avec `sleep(temps en millisecondes)` ou `xpause(temps en microsecondes)`;
- réaliser des simulations en temps réel avec `realtimeinit` (qui permet de fixer l'unité de temps) et `realtime`. Le premier appel à `realtime` fixe l'origine des dates, les appels suivants obligent Scilab à attendre qu'une date soit passée pour continuer.

Cet extrait provient du livre Scilab : De la théorie à la pratique - I. Les fondamentaux
écrit par Philippe Roux - © 2013 Éditions D-BookeR

Testez ces différentes commandes avec les exemples suivants :

```
-->// chronométrage avec tic() et toc()

-->tic()

-->sleep(1000)    //1000ms=1seconde

-->toc()
 ans  =

    1.006


-->tic()

-->xpause(200000)    //200000micros=0.2 secondes

-->toc()
 ans  =

    0.203

-->// temps CPU avec timer

-->timer();

-->sleep(1000)    //1000ms=1seconde

-->timer()
 ans  =

    0.0780005

-->timer();

-->xpause(1000000)    //1000000micros=1 seconde

-->timer()
 ans  =

    0.0624004

-->//temps réel

-->realtimeinit(1)    // unité de temps 1 seconde

-->realtime(0)    // date fixée à t=0

-->tic()

-->realtime(2)    //attendre date t=2

-->toc()    //le chronomètre affichera 2 secondes
 ans  =
```

Cet extrait provient du livre Scilab : De la théorie à la pratique – I. Les fondamentaux
écrit par Philippe Roux – © 2013 Éditions D-BookeR

```
    2.003
```

Vous pouvez effectuer des calculs sur les dates, mais prenez garde qu'il existe plusieurs formats pour gérer les dates et autant de fonctions pour les manipuler.

- `clock` récupère la date sous la forme d'une liste à six paramètres `[année, mois, jour, heure, minute, seconde]` ;
- `datenum` récupère la date sous la forme du nombre de jours écoulés depuis le 1er janvier de l'an zéro ;
- `getdate` récupère la date sous forme d'un timestamp (nombre de secondes écoulées depuis le 1er janvier 1970) ou d'une liste à dix paramètres `[année, mois, semaine, jour Julien, jour de la semaine, jour du mois, heure, minute, seconde, millisecondes]` (plus complexe que celle renvoyée par `clock` !).

Une fois les dates récupérées, vous pouvez les traiter avec d'autres commandes :

- `datevec` qui convertit un timestamp en une liste correspondant à la date en question pour Scilab ;
- `weekday` qui calcule le jour de la semaine correspondant à un timestamp ;
- `eomday` qui calcule le dernier jour du mois d'une année donnée ;
- `etime` qui calcule l'écart (en secondes) entre deux dates données par des listes à six paramètres.

Pour finir, les fonctions suivantes affichent des dates ou des calendriers :

- `date` récupère la date sous forme de chaîne de caractères ;
- `calendar` affiche un calendrier mensuel ou annuel.

Attention *› Pour la fonction* `weekday` *le premier jour de la semaine est le dimanche, alors que pour la fonction* `calendar` *le premier jour de la semaine est le lundi. Sans arguments, ces fonctions renvoient les valeurs correspondant à la date courante, sinon elles renvoient les valeurs correspondant aux dates données en paramètres.*

Cet extrait provient du livre Scilab : De la théorie à la pratique – I. Les fondamentaux
écrit par Philippe Roux – © 2013 Éditions D-BookeR

Voici quelques exemples d'utilisation des fonctions précédentes :

```
-->calendar(1970,1)    //calendrier Jan. 1970
 ans  =


       ans(1)

 Jan. 1970

       ans(2)

    L Ma M Je V Sam Dim

       ans(3)

    0.     0.     0.     1.     2.     3.     4.
    5.     6.     7.     8.     9.     10.    11.
    12.    13.    14.    15.    16.    17.    18.
    19.    20.    21.    22.    23.    24.    25.
    26.    27.    28.    29.    30.    31.    0.
    0.     0.     0.     0.     0.     0.     0.

-->eomday(2012,2)    // dernier jour de Février 2012
 ans  =

    29.

-->d1=[1970 1 1 0 0 0]    // format de date scilab
 d1  =

    1970.    1.    1.    0.    0.    0.

-->t1=datenum(d1)    //numéro de la date d1
 t1  =

    719529.

-->[N,S]=weekday(t1)    // jour de la semaine de la date d1
 S  =

 Jeu.
 N  =

    5.

-->date()    //date actuelle
 ans  =

 25-Août-2014

-->d2=clock()    //liste scilab de la date actuelle
 d2  =

    2014.    8.    25.    12.    16.    24.999998
```

Cet extrait provient du livre *Scilab : De la théorie à la pratique – I. Les fondamentaux*
écrit par Philippe Roux – © 2013 Éditions D-BookeR

```
-->t2=datenum(d2)    //numéro de la date d2
 t2  =

    735836.51

-->datevec(t2)    // date correspondant à t2
 ans  =

    2014.    8.    25.    12.    16.    24.999998

-->etime(d1,d2)    //écart entre les dates d1 et d2
 ans  =

  - 1.409D+09

-->d=etime(d2,d1)    //écart entre les dates d2 et d1
 d  =

    1.409D+09

-->getdate(d)    // jour situé d secondes après le 1er Jan. 1970
 ans  =

    2014.    8.    35.    237.    2.    25.    13.    16.    24.
 0.9999983
```

4.4. Historique des commandes

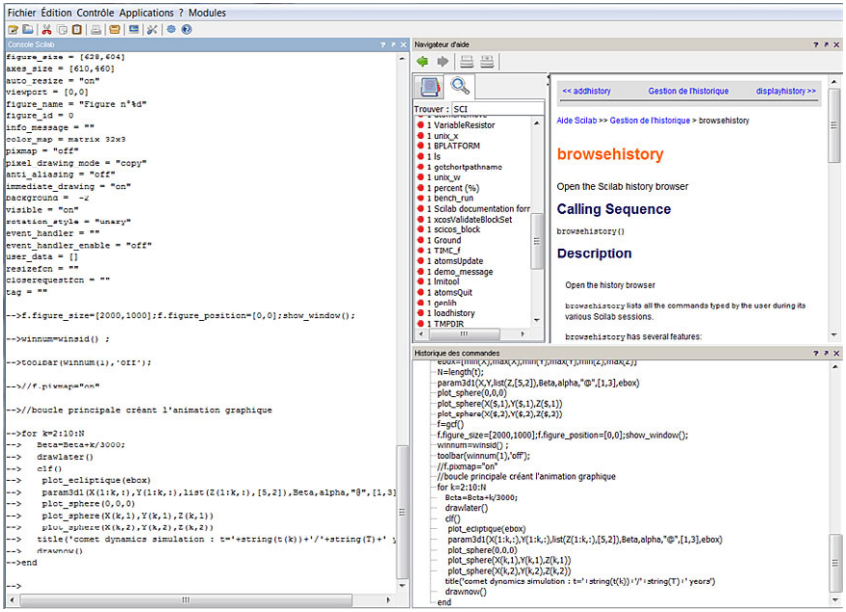
Scilab fournit un système de gestion de l'historique. Il permet d'enregistrer automatiquement les commandes saisies dans la console, ce qui permet de passer plus rapidement des tests à la rédaction d'un script. À l'aide des différentes fonctions, vous pourrez :

- effacer l'historique avec `resethistory` ;
- le sauvegarder dans un fichier avec `savehistory`, le recharger dans l'environnement Scilab avec `loadhistory` ;
- modifier des lignes de l'historique avec `addhistory` et `removelinehistory` ;
- récupérer l'historique dans une variable avec `gethistory` ;
- l'afficher dans le gestionnaire d'historique avec `browserhistory` ou dans la console avec `displayhistory`.

Vous pouvez aussi utiliser le gestionnaire d'historique (voir Figure 4.3) si vous préférez utiliser une interface graphique pour effectuer les opérations précédentes.

Cet extrait provient du livre Scilab : De la théorie à la pratique – I. Les fondamentaux
écrit par Philippe Roux – © 2013 Éditions D-BookeR

Figure 4.3 : Le gestionnaire d'historique



Pour finir, la fonction `diary` permet de stocker dans un fichier texte tout ce qui a été affiché dans la console (les commandes et leur résultat le cas échéant) entre deux appels à la fonction `diary`.

```
-->path=pwd();                  // répertoire courant

-->id=diary('scilab-base-diary.txt')    // ouverture du journal
 id  =

    2.

-->cd TMPDIR                    // répertoire temporaire de Scilab
 ans  =

 C:\cygwin\tmp\SCI_TMP_4880_

-->resethistory()               // efface l'historique

-->addhistory('ls')             // ajoute une ligne dans l'historique

-->gethistory()                 // récupère l'historique dans une variable
 ans  =
```

Cet extrait provient du livre Scilab : De la théorie à la pratique – I. Les fondamentaux
écrit par Philippe Roux – © 2013 Éditions D-BookeR

```
!// -- 25/08/2014 12:16:24 -- //  !
!                                 !
!ls                               !

-->displayhistory()          // affiche historique
0 : // -- 25/08/2014 12:16:24 -- //
1 : ls

-->savehistory('essai.txt')  // sauve l'historique dans un fichier

-->dir('essai.txt')          // le fichier est bien créé dans le
 répertoire courant
 ans  =

essai.txt

-->browsehistory()           // ouvre le gestionnaire de l'historique

-->cd(path);                 // retour au répertoire de départ

-->diary(id,'close')         //fermeture du journal
```

Attention *› Lorsqu'on exécute un programme Scilab stocké dans un fichier (comme ceux présentés au chapitre Aperçu de Scilab) en le lançant à l'aide des raccourcis de l'interface graphique (voir la vidéo de la Figure 3.5), les commandes exécutées ne sont pas stockées dans l'historique, y compris l'appel à la commande* `exec` *qui apparaît dans la console.*

Cet extrait provient du livre Scilab : De la théorie à la pratique – I. Les fondamentaux
écrit par Philippe Roux – © 2013 Éditions D-BookeR