

Java

Interface

Applet

University of Geneva
www.miralab.ch

Interface

2

Définition

C'est une collection de définitions de méthode sans implémentations

Une interface peut aussi déclarer des constantes

Syntaxe :

```
qualificatif interface I extends J
{ }
```

qualificatifs

absent : l'interface n'est utilisable que par les classes du même *package*

public : l'interface est utilisable par toutes les classes,
! une seule interface *public* par fichier

University of Geneva
www.miralab.ch

LCI
Introduction au langage C

Intérêt

3

Permet de savoir qu'une classe
possède ces méthodes
se comporte de telle manière

Relation avec une classe

```
class C implements I //, K, ...  
{  
    //implémentations de toutes  
    //les méthodes de I  
}
```

Autre utilisation des Interfaces

4

Masquage de classe d'implémentation

Certaines classes Java sont cachées derrière
une interface commune

Permet de créer des classes liées à différents
systèmes d'exploitation (donc de Machine
Virtuelles) sans perdre la portabilité

Autre utilisation des Interfaces

5

Interface vide

Permet de créer une catégorie de classes :
chaque classe implémentant ce type
d'interface appartient à telle ou telle catégorie

Pour tester si la classe d'un objet appartient à
une catégorie, il suffit d'utiliser l'opérateur
instanceof() avec le nom de l'interface

Exemple : *interface Cloneable*

Autre utilisation des Interfaces

6

Déclaration d'un ensemble de constantes

Utilisées dans des classes sans lien d'héritage

```
interface ConstantesCommunes
{ //Déclaration des constantes A, B et C
}

class Classe1 extends SuperClasse1 implements
    ConstantesCommunes
{ // Les constantes A, B et C sont utilisables
  dans la classe Classe1
}

class Classe2 extends SuperClasse2 implements
    ConstantesCommunes
{ // Les constantes A, B et C sont utilisables
  dans la classe Classe2
}
```

Exemple d'interface

```
public interface appareilÉlectrique
{
    /** teste si l'appareil est enclenché */
    public boolean estEnclenché();
    /** on appelle cette méthode lorsque l'on
     * branche l'appareil dans une source de
     * courant active, avec true, ou false
     * si la source est inactive */
    public void alimenté(boolean alim);
}
```

Exemple de classe

```
class lampe implements appareilÉlectrique
{
    private boolean allumée = false;
    public boolean estEnclenché()
    { return allumée; }
    public void alimenté(boolean alim)
    { allumee = alim; }
}
```

Différences avec classe abstract

Une *interface* ne peut pas définir de variable
seulement des constantes

Une classe ne peut hériter que d'une seule super
classe (*abstract* incluse), mais implémenter
plusieurs *interfaces*

On peut implémenter des méthodes (non
abstract) dans une classe *abstract*, aucune
implémentation acceptée dans une *interface*

Héritage des interfaces

Attention une classe qui implémente une interface
qui hérite d'une super-interface doit implémenter **toutes**
les méthodes définies dans ces interfaces

```
interface A
{ //Déclarations méthodes A
}
interface B extends A
{ //Déclarations méthodes B
}
class Classe1 implements B
{ // Définitions des méthodes de B et A
}
```

Interface comme type d'objet

11

Le nom d'une interface peut être utilisé comme type d'objet

Seules des instances de classes implémentant cette interface peuvent être assignées à des variables de ce type

```
public interface A
{ //Déclarations méthodes A
}
class ClasseA implements A
{ public void méthode(A objetdetypeA)
{   objetdetypeA = new ClasseA();
}
}
```

Tout objet de type *interface X* sont garantis d'avoir les méthodes déclarées dans cette *interface X*

12

```
public interface A
{ public int méthodeA(A objetdetypeA);
}
class ClasseA implements A
{ public void méthodeC(A objetdetypeA)
{   objetdetypeA.méthodeA();
}
public int méthodeA(A objetdetypeA)
{ //... }
}
```

Application & Applet

University of Geneva
www.miralab.ch

Rappel

14

Application

Programme indépendant

Point d'entrée du programme

```
public static void main(String [] args)
```

Applet

Programme téléchargeable

Exécuté automatiquement quand ils sont
intégrés à l'intérieur de pages HTML

University of Geneva
www.miralab.ch

LCI
Introduction au langage C

Applet Java

15

La classe d'une applet doit

- dériver de la classe *Applet*
- être *public*
- avoir un constructeur *public* sans paramètre

```
public class monApplet extends Applet
{
    public monApplet() {}
}
```

University of Geneva
www.miralab.ch

LCI
Introduction au langage C

Pour l'exécution il faut 2 fichiers

16

L'exécutable de l'applet
monApplet.class

Le fichier .html qui décrit l'applet au navigateur

tag html :

```
<APPLET CODE=monApplet.class ...>
</APPLET>
```

University of Geneva
www.miralab.ch

LCI
Introduction au langage C

Exemple Hello World

17

```
import java.applet.Applet;
import java.awt.Graphics;
public class AppletHelloWorld
    extends Applet
{ public void paint(Graphics g)
  { g.drawString("Applet Hello world !",
                 20, 20);
  }
  public static void main(String [] args)
  { System.out.println("Applet Hello World
  !"); }
}
```

Exemple fichier .html

18

```
<HTML>
<HEAD>
<TITLE> Applet HelloWorld </TITLE>
</HEAD>
<BODY>   Résultat :
<APPLET CODE="AppletHelloWorld.class"
  WIDTH=150 HEIGHT=25> </APPLET>
</BODY>
</HTML>
```

Lancement

Avec appletviewer

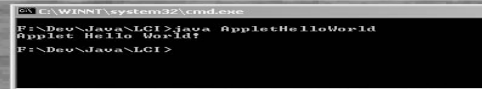
appletviewer AppletHelloWorld.html

Avec java

java AppletHelloWorld

Avec Netscape (IE, etc)

AppletHelloWorld.html



Passage de paramètres

20

Des paramètres peuvent être passés à une applet grâce aux tags html

```
<PARAM NAME="Param" VALUE="ValueParam">
```

inclus après <APPLET ...>

Chaque occurrence <APPLET> crée une instance de l'applet

Pour retrouver ces paramètres dans le code de l'applet, on fait appel à la méthode

```
String getParameter(String name)
```

21

Syntaxe complète des tag html

```
<APPLET CODE=ClasseApplet
  WIDTH=largeur HEIGHT=hauteur
  CODEBASE="répertoire" ALT="Ca marche
pas" NAME="NomApplet"
  ALIGN=alignement
  ARCHIVE="fichier.jar"> Message
(html) pour clients non
java</APPLET>
```

ClasseApplet correspond au fichier *.class*
WIDTH et **HEIGHT** définissent la largeur et la
 hauteur de la zone où sera affichée l'applet

University of Geneva
 www.miralab.ch

LCI
 Introduction au langage C

22

Syntaxe complète des tag html

CODEBASE (option) définit le chemin d'accès
 aux classes utilisées par l'applet

Par défaut le chemin d'accès est le répertoire d'où
 provient le fichier **HTML**

Le chemin spécifié par **CODEBASE** peut être relatif
 au répertoire courant du fichier HTML, ou être une
 URL désignant un chemin sur un serveur différent

ALT (option) définit la chaîne de caractères à
 afficher quand l'applet ne peut être exécutée

Quand un navigateur n'autorise pas Java, il écrira
 cette chaîne

University of Geneva
 www.miralab.ch

LCI
 Introduction au langage C

Syntaxe complète des tag html

NAME (option) définit un nom pour l'applet

Utilisé quand vous recherchez une applet par son nom

ALIGN (option) définit l'alignement horizontal de l'applet dans la page HTML

LEFT, RIGHT ou MIDDLE

ARCHIVE (option) définit le fichier .JAR

qui rassemble les classes, les images et autres fichiers qu'utilise l'applet

JAR = *Java ARchive*

Méthodes outre passables

D'une classe d'applet

`init()` appelée à la création, similaire au `main()`

`start()` appelée à son affichage

`stop()` appelée à son effacement

`destroy()` appelée à sa destruction

Classe *Applet*

25

Dérive des classes

Panel, *Container*, *Component* du package `java.awt`

La méthode *paint()* utilisée dans l'exemple précédent est une redéfinition de la méthode de *Component*

Une autre méthode souvent redéfinie dans les applets provient de *Container*

add() qui permet par exemple d'ajouter des boutons sur une page web

Autres méthodes utiles

26

Classe *Applet*

```
public String getAppletInfo()
public String [ ][ ] getParameterInfo()
public void resize (int width, int height)
public URL getDocumentBase()
public URL getCodeBase()
public AppletContext getAppletContext()
public boolean isActive()
public void showStatus(String msg)
public Image getImage(URL url, String name)
AudioClip getAudioClip(URL url, String name)
public void play(URL url, String name)
```

Sécurité

27

Le gestionnaire de sécurité

SecurityManager de la Machine Virtuelle
est très restrictif pour éviter toute intrusion
dans le système où fonctionne le
navigateur

Aucun accès au système de fichiers local

Possibilité d'accéder uniquement à la machine
sur lequel est hébergé le fichier de la classe
de l'applet

Sécurité

28

Impossibilité

de lancer des applications locales

de définir des méthodes native et d'utiliser des
bibliothèques du système

Accès limité aux propriétés du système de la
Machine Virtuelle

Les fenêtres créées par une applet (classe
Window) comportent un bandeau indiquant
que la fenêtre a été créée par l'applet

Contrôle du niveau de sécurité

29

appletviewer et certains navigateurs
permettent de modifier les paramètres du
gestionnaire de sécurité

Pour autoriser certaines opérations

Pour réaliser des applets accessibles
publiquement sur Internet

La conception doit considérer le niveau de sécurité
maximum des navigateurs

Tests d'applet

30

Sur certaine Machine Virtuelle Java

tant que vous ne quittez pas l'interpréteur

java, *appletviewer* ou navigateur

les classes déjà chargées restent en mémoire

et ne sont pas rechargées quand vous relancez la
même applet

il faut quitter l'interpréteur à chaque fois que vous
changez une des classes de votre programme

sinon vous aurez l'impression que vos modifications
n'ont pas été prises en compte !...

Exemple

31

Applet "brouillon pour dessiner"

```
<title>Applet</title>
```

Exemple:

```
<hr>
```

```
<applet code=Interaction.class width=250  
height=150>
```

```
</applet>
```

```
<hr>
```

32

```
import java.awt.Graphics;
import java.awt.Event;
public class Interaction
    extends java.applet.Applet
{ int x0,y0;
  public boolean handleEvent(Event e)
  { switch(e.id)
    { case Event.MOUSE_DRAG:
      Graphics g= getGraphics();
      g.drawLine(x0,y0,e.x,e.y);
      case Event.MOUSE_DOWN:
        x0=e.x; y0=e.y; break;
      default:
        return super.handleEvent(e);
    }
  }
  return true;
}
```



Exemple

33

Applet "poursuite du curseur"

```
<title>Applet</title>
```

Exemple:

```
<hr>
```

```
<applet code=TrackSouris.class width=250
      height=150>
```

```
</applet>
```

```
<hr>
```

```
import java.awt.Graphics;
import java.awt.Event;
import java.awt.event.*;
public class TrackSouris
    extends java.applet.Applet
    implements MouseMotionListener
{ int sourisX, sourisY;
  long quand;
  public void init() {
    this.addMouseMotionListener(this); }
  public void mouseMoved(MouseEvent e){
    quand=e.getWhen(); sourisX=e.getX();
    sourisY=e.getY(); repaint(); }
  public void mouseDragged(MouseEvent e) {}
  public void paint(Graphics g) {
    g.drawString(quand+"("+sourisX+", "
                +sourisY+")" ,10,20); }
}
```



34

Exemple avec paramètres

35

```
<title>Fraternite.html</title>
```

Exemple: <hr>

```
<applet code=Fraternite.class width=350
height=250>
```

```
  <PARAM NAME="Fraternité"
    VALUE="Fratricidité">
```

```
</applet>
```

```
<hr>
```

University of Geneva
www.miralab.ch

LCI
Introduction au langage C

```
import java.awt.Graphics;
import java.awt.Color;
public class Fraternité
    extends java.applet.Applet
{ String fraternité;
  public void init()
  { fraternité = getParameter("fraternité"); }
  public void paint(Graphics g)
  { g.setColor(Color.blue);
    g.drawString("Liberté", 50, 60);
    g.setColor(Color.white);
    g.drawString("Egalité", 50, 90);
    g.setColor(Color.red);
    if(fraternité != null)
      g.drawString(fraternité, 50, 120);
  }
}
```

36

University of Geneva
www.miralab.ch

LCI
Introduction au langage C

Résumé

37

Interface	Applets
Utilité et utilisation	Lancement de l'exécution
Héritage	Passage de paramètres
Type	Points d'entrées <i>init(), start(), stop(), destroy()</i>
	Classe <i>Applet</i> Sécurité et implications Tests d'applet Exemples d'Applet

University of Geneva
www.miralab.ch

LCI
Introduction au langage C