Université de Nice – Sophia Antipolis — Deug MASS MIAS SM 1re année
UEF1 : Informatique & Programmation — 2000–2001

TP N° 6

Du graphisme dans une *applet*

Ça va mieux en le révisant ... L'API (*Application Programming Interface*) de JAVA est l'ensemble de toutes les classes prédéfinies (comme `Math`) constituant la bibliothèque initiale de JAVA . On y trouve en particulier deux bibliothèques graphiques, l'AWT (*Abstract Windowing Toolkit*) que nous utiliserons, et son extension SWING que nous laisserons de côté. L'API est organisée en *packages* (paquetages ou bibliothèques en français), par exemple le package `java.awt`, chaque paquetage contenant de nombreuses classes (par exemple `java.awt.Color`).

Une *applet* est un programme JAVA destiné à être exécuté dans la fenêtre d'un navigateur Internet (EXPLORER, NETSCAPE). Pour construire une applet, il suffit d'étendre la classe `java.applet.Applet` de l'API, en redéfinissant la méthode `paint(Graphics g)` qui exprime comment va s'afficher l'applet. Le lancement de l'applet est effectué par le navigateur lorsqu'il lit la section `<applet>...</applet>` d'une page HTML [1]. Il est dit dans cette section où se trouve la classe JAVA correspondante, ainsi que les dimensions de la zone rectangulaire réservée à l'applet dans la page.

1 Ne lancez pas REALJ ... surfez dans la doc.

La capacité de naviguer rapidement au sein de la documentation de l'API de JAVA est absolument indispensable pour programmer de manière avancée (notamment graphique) en JAVA . Il est donc vital d'acquérir vite le réflexe d'aller butiner dans les classes à la recherche de la méthode adéquate pour effectuer une tâche, ou pour connaître les types des arguments attendus par une méthode.
We do hope you will be able to read it...

Exercice 1) La documentation de l'API est un ensemble de pages HTML accessible à partir du menu *Démarrer*, à la rubrique Java : choisissez **Documentation API Java**, et ces pages s'ouvriront automatiquement dans *Internet Explorer*. Vous notez un découpage en 3 zones. En haut à gauche, la fenêtre des *packages*. En bas à gauche, la fenêtre des *classes*. Sur la droite, la fenêtre détaillée des descriptions.

Où se trouve la classe `Math` ? Trouvez-la dans la fenêtre inférieure gauche, cliquez sur son nom et vous la verrez à droite. Tiens, elle se trouve dans le package `java.lang` puisque son nom complet est en réalité `java.lang.Math` [2]. En fait, ce package `java.lang` est tellement important qu'il est disponible d'emblée, sans qu'il soit besoin de le demander explicitement par une directive `import`. La classe est `public final` : elle est publique et non modifiable.

a) Cherchez comment vous nommerez la constante e base des logarithmes népériens si vous en avez besoin dans un programme. Quel est son type ? Pourquoi est-elle déclarée en `static` ?

b) Quel est le nom de la méthode qui arrondit un *float* vers un *int*, par exemple 6.74 en 7 ? Que doit-on faire pour arrondir un `double` ?

c) Quel est le nom complet de la classe `Console` du package `unsa` ? ◇

1. HTML = HyperText Markup Language, le langage de description des pages du Web.
2. Ne confondez pas avec le package `java.math` qui contient d'autres outils plus spécialisés et d'usage moins fréquent.

Exercice 2) Occupons-nous maintenant un peu de l'AWT. Cherchez dans la fenêtre supérieure gauche le package java.awt[3]. Faites apparaître la classe `Color` dans la partie droite. C'est une sous-classe de `Object` : une *couleur* est donc bien un *objet.*

a) Quel est le nom de la constante *couleur bleue*? C'est une *constante de classe.* Pourquoi?

b) Comment auriez-vous défini cette couleur avec le constructeur `Color`(...)? Le système RGB (Red-Green-Blue) considère que toute couleur s'obtient en mélangeant ces trois couleurs de base avec des proportions adéquates.

c) Comment s'obtient le gris? le jaune?

d) Etant donnée une variable `couleur` de type `Color`, comment obtenir sa composante verte? Quel sera le type de cette composante? ◇

Exercice 3) Toujours dans l'AWT, la classe `Graphics`. Un objet de la classe `Graphics` (par exemple celui qui est en paramètre de la méthode `paint`(...) d'une applet) se nomme un **contexte graphique**. C'est à un tel objet que l'on s'adresse pour dessiner un cercle, un segment, changer la couleur du tracé, etc.

Lorsque l'on dessine dans une fenêtre graphique, on le fait dans un système d'**axes orthonormés**. En JAVA, **l'origine du repère** est le coin supérieur gauche de la fenêtre, **l'axe Ox** est horizontal orienté **de gauche à droite**, et **l'axe Oy** est vertical orienté **de haut en bas** contrairement à l'axe *Oy* des matheux, donc attention !

En supposant que la variable `g` fait référence à un contexte graphique, quelles seraient les instructions qui dessineraient en bleu un cercle de rayon 30 et dont le centre a pour coordonnées $(25, 40)$? ◇

2 Programmer une applet graphique simple

Exercice 4) Ouvrez un nouveau projet `TP6.jpr` dans REALJ.

a) Sauvez dans un fichier `AppletCours6.java` le source de l'applet vue en cours. Compilez ce fichier sans l'exécuter, simplement pour obtenir le fichier `AppletCours6.class`.

b) Toujours dans REALJ, demandez `New HTML file`. Tapez le texte de la page HTML vue en cours, qui avait pour but de lancer l'applet, puis sauvez ce texte HTML dans un fichier `AppletTP6.html`. N'essayez pas de le compiler ou de l'exécuter, ce n'est pas un texte source en JAVA !

3. Notez que les noms de packages sont toujours en minuscules, contrairement aux noms des classes, qui débutent toujours par une Majuscule.

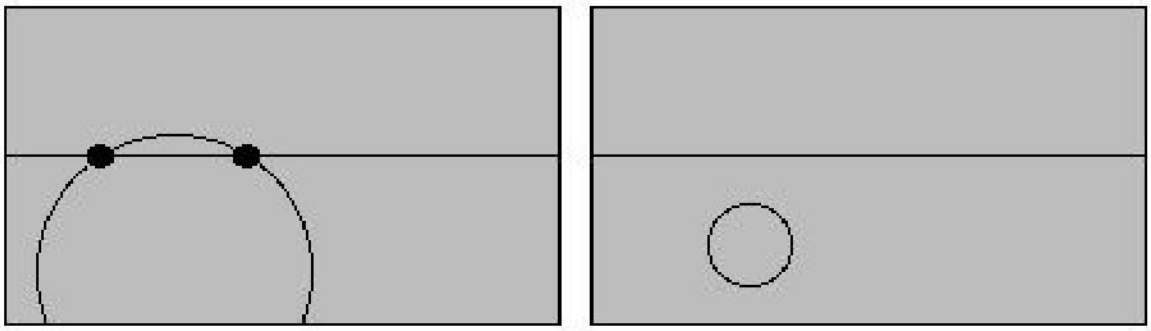
c) Il y a deux manières de lancer l'exécution de l'applet, soit en passant par un *navigateur Web* (EXPLORER, NETSCAPE), soit par le biais de l'*Applet Viewer* contenu dans le JDK[4]. Le plus simple consiste à rester dans REALJ et, avec le fichier HTML en cours d'édition, demander *Run Applet* dans le menu *Build* (il se peut que l'équivalent-clavier Shift-F5 ne fonctionne pas...). REALJ va automatiquement lancer le programme externe *Applet Viewer* qui se contentera d'extraire la section :

```
<applet code="..." width=... height=...></applet>
```

avec les informations qu'elle contient, à savoir l'endroit où se trouve le fichier `Xxxx.class` et les dimensions de la fenêtre[5]. Le reste de la page HTML est ignoré. Essayez et ... mettez au point jusqu'à ce que tout fonctionne normalement. Attention, si votre fenêtre REALJ est en mode zoom, il se peut que l'*Applet Viewer* soit caché derrière, retrouvez-le en diminuant la fenêtre REALJ. Dans le menu de l'*Applet Viewer*, vous trouverez le moyen de relancer l'applet directement.

d) Fermez l'*Applet Viewer*, mais pas REALJ. Ouvrez NETSCAPE ou EXPLORER, et chargez la page HTML. Vous verrez alors tout le contenu de cette page dans la fenêtre du navigateur, et en particulier la petite fenêtre réservée à l'applet. Lorsque vous écrirez des pages HTML en Bureautique, sachez donc que vous pourrez y inclure des applets, écrites par vous si possible, ou bien téléchargeables sur le Web. Mais dans la phase de programmation, l'utilisation de l'*Applet Viewer* est plus confortable. ◇

Exercice 5) En modifiant le programme ci-dessus, écrivez une classe `AppletExoTP6` réalisant une applet de dimensions 300x200 et de fond gris. Une ligne horizontale bleue la traversera au milieu ; un cercle jaune de centre et de rayon raisonnablement aléatoires doit aussi être dessiné. Calculez les coordonnées des points d'intersection de la droite et du cercle, et *s'ils existent*, affichez-les sous la forme de petits disques *pleins* de couleur rouge. Deux exemples :



◇

4. Sur Macintosh, utiliser *Apple Applet Viewer* qui se trouve dans le dossier *MacOS Runtime for Java*. Pour plus de détails sur la programmation en JAVA sur Mac, voir :

`http://deptinfo.unice.fr/~roy/Java/JavaSurMac/JavaSurMac.html`

5. On peut passer d'autres informations, comme le répertoire du fichier `.class`, ou encore des valeurs à fournir à l'applet, mais nous nous limiterons aux paramètres `code`, `width` et `height`.