LE GOUVERNEMENT
DU GRAND-DUCHÉ DE LUXEMBOURG
Ministère de l'Éducation nationale,
de l'Enfance et de la Jeunesse

Enseignement secondaire général
Classes supérieures
Division technique générale
Section informatique
PROGR / Programmation
Programme
2GIN

Langue véhiculaire :	français
Nombre de leçons :	4
Nombre minimal de devoirs par trimestre :	2
Dernière mise à jour par la CNES :	24/05/2021

Remarques générales

1. Vu l'aspect pratique du cours, celui-ci doit avoir lieu entièrement dans une salle d'informatique avec un poste par élève. L'utilisation d'une tablette ne se substitue pas à un ordinateur dans ce cadre.
   Le nombre d'élèves par auditoire ou par enseignant doit être limité à 15.
   Il est recommandé d'organiser le cours à raison de blocs de deux leçons consécutives.
2. Ce cours peut être suivi et conclu avec succès uniquement si les élèves peuvent se préparer et s'exercer sur un ordinateur de bureau en dehors des cours. L'accès à un ordinateur qui permet d'installer le JDK Java (donc tournant sous MacOS, Windows ou Linux) est nécessaire.
3. Le programme est axé sur quelques projets définis, dans le cadre desquels les élèves apprennent à manipuler les différents éléments énumérés dans la partie "Détails".
4. Les applications à interface graphique développées dans le cours se limitent à un seul formulaire.
5. Les algorithmes plus complexes (par exemple : boucles imbriquées) sont développés et représentés sous forme de structogrammes. Les élèves doivent être capables de lire et interpréter les structogrammes ainsi que de réaliser les algorithmes correspondants en Java.
6. Les classes réalisées dans le cours ainsi que et les relations (héritage et agrégation) entre ces classes sont représentées sous forme de diagrammes UML. A la fin du cours,

les élèves doivent être capables de lire, d'interpréter et de réaliser les diagrammes de classe UML.
7. Les algorithmes communément appelés « problèmes type », proposés et publiés par la commission nationale des programmes, font partie intégrante du programme en informatique de la division technique générale.
8. Dans le cas idéal, chaque trimestre se termine par un projet de synthèse.

A. Objectif

L'élève est capable de concevoir, de réaliser et d'adapter des applications sous un environnement graphique et événementiel selon le modèle « MVC ». A cet effet il connaît les manipulations essentielles du langage de programmation et fait appel à la pensée informatique pour résoudre des problèmes sous forme de projets mettant en œuvre une approche orientée objets.
Vu que le modèle MVC (Model-View-Controller) doit être suivi de manière très rigoureuse tout au long du cours, l'utilisation du logiciel Unimozer pour le développement de la partie « modèle » est vivement recommandée. Il en est de même pour le chapitre relatif à l'héritage.
La première partie du cours consiste à développer l'interface graphique (« vue » et « contrôleur ») pour des classes (« modèles ») qui ont été ou auraient pu être développées en 3GIG. De ce fait, cette partie sert aussi à approfondir et à perfectionner les compétences y acquises.

Compétences ciblées	Contenus	Durée
1. Savoir lire et interpréter des structogrammes		-
2. Savoir lire, interpréter et réaliser des diagrammes de classes UML	- structure générale - symboles de visibilité - relations (héritage et agrégation)	-
3. Connaître et savoir appliquer le modèle MVC (Model-View-Controller) Connaître la notion de paquets en Java	- concepts - avantages - mot clé new - la valeur null - structure des paquets - mot clé import	2
4. Connaître et savoir utiliser les opérations de base sur les chaînes de caractères	- classe *String*	1
5. Être capable de réaliser des conversions entre types simples	- notion de auto-boxing - les classes enveloppes (*Integer*, *Long*, *Float*, *Double*, *Boolean*)	1
6. Savoir créer des applications graphiques simples se basant sur le modèle MVC	- classes Swing (JFrame, JButton, JLabel, JTextField, JSlider, JPanel, JList) - classes AWT (ActionListener, ActionEvent)	16
7. Connaître et savoir manipuler des listes	- -classe « ArrayList » / « Vector » - notions de « templates » - classe Swing « JList » - recherche ou calcul du maximum, minimum, somme et moyenne des éléments d'une liste - recherche linéaire et dichotomique - ri par sélection directe - projet de synthèse	20
8. Savoir dessiner sur un descendant de la classe « JPanel »	- classe Swing « JPanel » - classes AWT « Graphics » et « Color » - projet de synthèse : traceur de fonctions polynomiales en insistant sur la distinction entre coordonnées dans le plan mathématique et le plan graphique	16
9. Comprendre et savoir utiliser un chronomètre	- classe « javax.swing.Timer » : constructeur, start(), stop(), isRunning(), setDelay(...), getDelay() - méthode du "bouton caché" - méthode par classe interne	4
10. Savoir accéder la date et le temps actuel	- utiliser java.time.LocalTime, java.time.LocalDate, java.time.LocalDateTime	4

Compétences ciblées	Contenus	Durée
	- programmer une montre digitale et une montre analogique (→ dessin)	
11. Savoir mesurer le temps et comparer l'efficacité de quelques algorithmes standard	- System.nanoTime() - algorithmes à comparer : recherches linéaire et dichotomique, tris par sélection, insertion, propagation, quicksort (donné), Collection.sort()	8
12. Faire réagir une classe aux événements de la souris	- classes « MouseListener », « MouseMotionListener », « MouseEvent » - accès aux coordonnées de la souris - savoir déplacer un objet	10
13. Savoir mettre en œuvre certains dialogues standard	- classe « JColorChooser »	1
14. Connaître la notion de « agrégation » Connaître et savoir mettre en œuvre l'héritage	- mot clé « extends » - redéfinition de méthodes - redéfinition du constructeur - mot clé « super » - polymorphisme et « late binding » - mot clé « abstract » (pour méthodes et classes) - opérateur « instanceof »	15
15. Connaître et savoir employer les classes, attributs et méthodes finales et statiques	- les mots clés static et final, - leur utilité et leur emploi	4
16. 16. Connaître et savoir utiliser les interfaces	- mot clé « interface » - mot clé « implements »	1
17. Savoir réaliser une application de dessin vectoriel	- ajouter des figures (rectangle, ellipse, ligne) à l'aide de la souris - déplacer et supprimer des figures - modifier la couleur des figures - changer l'ordre de dessin des figures - sauvegarder un dessin dans un fichier - charger un dessin à partir d'un fichier	17
	TOTAL :	**120**

B. Détails

Package	Classe	Details	Remarques
javax.swing	JFrame	Méthodes - setTitle(...) / getTitle() - getContentPane.getWidth() / getContentPane.getHeight() NetBeans Object Inspector Property - -title	
	JButton JLabel JTextField	Méthodes - setText(...) / getText() - setVisible(...) - setEnabled(...) Événement - actionPerformed NetBeans Object Inspector Property - icon	- le libellé « JLabel » peut aussi être utilisé pour visualiser des images via la propriété « icon » de l'inspecteur objet de NetBeans (le composant « JTextField » ne possède pas de propriété « icon »).
	JButton	- doClick()	
	JTextField	- selectAll()	
	JComponent	- requestFocus()	
	JSlider	Méthodes - setMinimum(...) / getMinimum() - setMaximum(...) / getMaximum() - setValue(...) / getValue() Événement - stateChanged	
	JPanel	Méthodes - setBackground(...) / getBackground() - getWidth() / getHeight() - paintComponent(...) - repaint() - setVisible(...) - getGraphics()	- « JPanel » est utilisé pour regrouper d'autres composants visuels et pour réaliser des dessins.
	JList	Méthodes - setListData(...) - getSelectedIndex(...) / setSelectedIndex() Événement - valueChanged NetBeans Object Inspector Properties - model - selectionMode : SINGLE	- *JList* est utilisé surtout pour afficher le contenu d'une liste *ArrayList*.
	Timer	Constructeur - Timer(int,ActionListener) Méthodes - setDelay(...) / getDelay() - start(), stop(), isRunning()	
	JColorChooser	Méthode	

Package	Classe	Details	Remarques
		- showDialog(...)	
java.awt	Graphics	Méthodes - drawLine(...) - drawOval(...) / fillOval(...) - drawRect(...) / fillRect(...) - drawString(...) - setColor(...) / getColor()	
	Color	Constructeurs - Color(int r, int g, int b) - Color(int r, int g, int b, int alpha)	- voir aussi les constantes de cette classe
	Point	Constructeurs - Point(...) Attributs (publics) - x et y Méthodes - getX(), getY() - getLocation(), setLocation(...)	
java.awt.event	ActionListener		
	ActionEvent		- Ce type d'objet est uniquement utilisé dans les méthodes de réaction ajoutées de manière automatique à l'aide de NetBeans.
	MouseListener	Événements - MouseClicked - MousePressed - MouseReleased	-
	MouseMotionListener	Événements - MouseDragged - MouseMoved	-
	MouseEvent	Méthodes - getButton() - getClickCount() - getPoint() / getX() / getY() - isShiftDown(), isAltDown(), isControlDown() Constantes BUTTON1, BUTTON2 et BUTTON3	-
java.util	ArrayList	Méthodes - add(...) - clear() - contains(...) - get(...) - indexOf(...) - remove(...) - set(...) - size() - toArray() - isEmpty()	- voir aussi la notion de « templates » - `Object[] toArray()` est employé uniquement pour passer les contenus d'une liste à la méthode setListData(...) d'une « JList ».
java.lang	String	Méthodes - equals(...) - compareTo(...) - contains(...)	

Package	Classe	Details	Remarques
		- valueOf(...)	
java.lang	Integer, Long, Float, Double, Boolean	Méthode - equals(...) / compareTo(...) - valueOf(...)	

C. Logiciels à utiliser :

Environnement de développement :

- Unimozer (http://unimozer.fisch.lu)
- NetBeans (http://netbeans.org)

D. Liens importants

- Ressources concernant le cours d'introduction à la programmation orientée objet
    - http://java.cnpi.lu/
- JavaDoc
    - https://docs.oracle.com/en/java/javase/
    - http://docs.oracle.com/javase/tutorial

E. Évaluation trimestrielle :

Les devoirs en classe peuvent se présenter sous différentes formes :

- devoir pratique sur ordinateur,
- devoir écrit,
- projet ou partie d'un projet,
- contrôle continu (maximum 1/6 de la note trimestrielle).