



Enseignement secondaire technique

Régime technique

Division technique générale informatique

Cycle supérieur

PROGR / Programmation

Programme

2GIN

Langue véhiculaire : français

Nombre minimal de devoirs par trimestre : 2

Remarques :

- Vu l'objectif "orienté-actions" et l'aspect pratique du cours, celui-ci doit avoir lieu entièrement dans une salle d'informatique. Les travaux pratiques se feront sur les ordinateurs à raison d'**un élève par poste**. Afin d'assurer l'acquisition des compétences visées, il s'avère essentiel de prévoir des **groupes de 12 élèves au maximum** et d'organiser le cours à raison de **blocs de deux leçons consécutives**.
- 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".
- Les applications à interface graphique développées dans le cours se limitent à un seul formulaire.
- 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.
- 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.
- Les algorithmes communément appelés « algorithmes connus », proposés et publiés par la commission nationale des programmes pour les classes 2GIG et 1GIG, font partie du programme de la 2GIN.
- 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 Unimozzer 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 <i>et dichotomique</i> - tri 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.LocalDate, java.time.LocalDateTime - programmer une montre digitale et une montre analogique (→ dessin)	4



Compétences ciblées	Contenus	Durée
11. Savoir mesurer le temps et comparer l'efficacité de quelques algorithmes standard	<ul style="list-style-type: none">- 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	<ul style="list-style-type: none">- classes « MouseListener », « MouseMotionListener », « MouseEvent »- accès aux coordonnées de la souris- savoir déplacer un objet	10
13. Savoir mettre en œuvre certains dialogues standard	<ul style="list-style-type: none">- classe « JColorChooser »	1
14. Connaître la notion de « agrégation » Connaître et savoir mettre en œuvre l'héritage	<ul style="list-style-type: none">- 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 <i>static</i> et <i>final</i> , leur utilité et leur emploi	4
16. Connaître et savoir utiliser les interfaces	<ul style="list-style-type: none">- mot clé « interface »- mot clé « implements »	1
17. Savoir réaliser une application de dessin vectoriel	<ul style="list-style-type: none">- 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 / Interface	Details	Remarques
javax.swing	JFrame	<u>Méthodes</u> - setTitle(...) / getTitle() - getContentPane.getWidth() / getContentPane.getHeight() <u>NetBeans Object Inspector</u> <u>Property</u> - title	
	JButton JLabel JTextField	<u>Méthodes</u> - setText(...) / getText() - setVisible(...) - setEnabled(...) <u>Événement</u> - actionPerformed <u>NetBeans Object Inspector</u> <u>Property</u> - 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	<u>Méthodes</u> - setMinimum(...) / getMinimum() - setMaximum(...) / getMaximum() - setValue(...) / getValue() <u>Événement</u> - stateChanged	
	JPanel	<u>Méthodes</u> - setBackground(...) / setBackground() - getWidth() / getHeight() - paintComponent(...) - repaint() - setVisible(...) - getGraphics()	- « JPanel » est utilisé pour regrouper d'autres composants visuels et pour réaliser des dessins.
	JList	<u>Méthodes</u> - setListData(...) - getSelectedIndex(...) / setSelectedIndex() <u>Événement</u> - valueChanged <u>NetBeans Object Inspector</u> <u>Properties</u> - model - selectionMode : SINGLE	- « JList » est utilisé surtout pour afficher le contenu d'une liste « ArrayList » / « Vector ».
	Timer	<u>Constructeur</u> - Timer(int, ActionListener) <u>Méthodes</u> - setDelay(...) / getDelay() - start(), stop(), isRunning()	



Package	Classe / Interface	Details	Remarques
	JColorChooser	<u>Méthode</u> - showDialog(...)	
java.awt	Graphics	<u>Méthodes</u> - drawLine(...) - drawOval(...) / fillOval(...) - drawRect(...) / fillRect(...) - drawString(...) - setColor(...) / getColor()	
	Color	<u>Constructeurs</u> - Color(int r, int g, int b) - Color(int r, int g, int b, int alpha)	- voir aussi les constantes de cette classe
	Point	<u>Constructeurs</u> - Point(...) <u>Attributs (publics)</u> - x et y <u>Méthodes</u> - getX(), getY() - getLocation(), setLocation(...)	
java.awt.event	ActionListener		
	ActionEvent		
	MouseListener	<u>Événements</u> -MouseClicked -MousePressed -MouseReleased	
	MouseMotionListener	<u>Événements</u> -MouseDragged -MouseMoved	
	MouseEvent	<u>Méthodes</u> -getButton() -getClickCount() -getPoint() -getX() / getY() <u>Constantes</u> BUTTON1, BUTTON2 et BUTTON3	
java.util	ArrayList	<u>Méthodes</u> -add(...) -clear() -contains(...) -get(...) -indexOf(...) -remove(...) -set(...) -size() -toArray()	- 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	<u>Méthodes</u> -equals(...) / compareTo(...) -contains() -valueOf(...)	
	Integer, Long,	<u>Méthode</u>	



Package	Classe / Interface	Details	Remarques
	Float, Double, Boolean	- equals(...) / compareTo(...) - valueOf(...)	

C. Logiciels à utiliser :

Environnement de développement :

- Unimozzer (<http://unimozzer.fisch.lu>)
- NetBeans (<http://netbeans.org>)

Éditeur de structogrammes :

- Structorizer (<http://structorizer.fisch.lu>)

D. Liens importants

- Ressources concernant le cours de programmation orientée objet Java
 - <http://java.cnpi.lu/>
- JavaDoc
 - <http://download.oracle.com/javase/8/docs/api/>

E. Lecture conseillée au titulaire :

Le site web des enseignants titulaires d'un cours d'informatique dans une classe de la division technique générale:
<http://cnpi-est.myschool.lu>

F. Évaluation trimestrielle :

L'évaluation trimestrielle peut se faire sous différentes formes :

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