```java
import java.lang.Object;
import java.io.*;
import javax.xml.parsers.*;
import org.xml.sax.*;
import javax.swing.tree.*;
import javax.swing.*;
import org.xml.sax.helpers.*;
import java.awt.*;

public class Aide{
        SAXParserFactory saxParserFactory;
        SAXParser saxParser;
        DefaultMutableTreeNode racine;
        DefaultTreeModel dtm;
        JTree tree;
        HandlerArbre handler;
        XMLReader r;
        JFrame maFen;
        JEditorPane jep;
        JButton btnFermer;
        AideListener ecouteur;

        private void creerArbo(File rep){
                Reader unReader;
                racine = new DefaultMutableTreeNode("Aide");

                File dossier[] = rep.listFiles();
                        for(int i=0; i<dossier.length; i++){
                        try{

                        saxParserFactory = SAXParserFactory.newInstance();
                        saxParser = saxParserFactory.newSAXParser();

                                File fichier = dossier[i];
                                unReader = new FileReader(fichier);
                                r = saxParser.getXMLReader();
                                handler = new HandlerArbre(fichier);
                                r.setContentHandler(handler);

                                r.parse(new InputSource(unReader));

                        }catch(Exception e){};

                        racine.add(handler.getBranche());
                }
        }

        public void chargerEditor(String url){


                try{

                                jep.setContentType("text/html");
                                //System.out.println("---> Passe");
                                jep.read(new FileReader(url), null);
                                jep.setEditable(false);
                        }catch(Exception ioe){/*System.out.println("Erreur");*/}
        }

        private JPanel centerPanel(){
                JPanel panel = new JPanel();
                panel.setLayout(new BorderLayout());

                jep = new JEditorPane();
                chargerEditor("./Aide/Conseil.htm");
                JScrollPane jsp = new JScrollPane(jep);
                panel.add(jsp, BorderLayout.CENTER);
```



```java
                return panel;
        }

        private JPanel bouton(){
                JPanel panel = new JPanel();
                btnFermer = new JButton("Fermer");

                panel.setLayout(new FlowLayout(FlowLayout.RIGHT));
                panel.add(btnFermer);

                return panel;
        }

        public Aide(File rep){

                // creation de l'arboresence
                creerArbo(rep);

                // ajout de l'arbre dans la fenetre
                dtm = new DefaultTreeModel(racine);
                tree = new JTree(dtm);

                maFen = new JFrame("Aide");
                JScrollPane jsp = new JScrollPane(tree);
                Container c = maFen.getContentPane();
                c.setLayout(new BorderLayout());
                c.add(jsp, BorderLayout.WEST);

                c.add(centerPanel(), BorderLayout.CENTER);
                c.add(bouton(), BorderLayout.SOUTH);

                // ecouteur
                ecouteur = new AideListener(this);
                btnFermer.addActionListener(ecouteur);
                tree.addTreeSelectionListener(ecouteur);

                maFen.setSize(400, 500);
                maFen.setVisible(true);
        }

}

class HandlerArbre extends DefaultHandler{
        DefaultMutableTreeNode branche;
        File fichier;
        //creer pile

        public HandlerArbre(File f){
                fichier = f;
                branche = new DefaultMutableTreeNode(fichier.getName());
        }

        public DefaultMutableTreeNode getBranche(){
                return branche;
        }

        public void startDocument(){
                //System.out.println("Les balises de " + fichier.getName() + " s
ont ");

        }

        public void endDocument(){
        }

        public void startElement(String uri, String locName, String qName, Attri
```

```
butes atr){
                int nb = atr.getLength();
                if (nb!=0){
                        for(int i=0; i<nb; i++){
                                if(atr.getQName(i).compareTo("name")==0){
                                DefaultMutableTreeNode feuille = new DefaultMuta
bleTreeNode(atr.getValue(i));
                                //DefaultMutableTreeNode feuille = new DefaultMu
tableTreeNode("toto");
                                branche.add(feuille);
                                //System.out.println("\t "+ atr.getValue(i));
                                }
                        }
                }
        }

        public void endElement(String uri, String locName, String qName){

        }

        public void characters(char[] chrs, int d, int n){

        }

}
```

```java
import java.io.*;
import javax.swing.tree.*;
import javax.swing.*;
import javax.swing.event.*;
import java.awt.*;
import java.awt.event.*;
import java.util.*;

public class AideListener implements ActionListener, TreeSelectionListener{
        Aide arbo;
        JButton fermer;
        JEditorPane jep;


        public AideListener(Aide a){
                arbo = a;
                fermer = a.btnFermer;
                jep = a.jep;
        }

        public void actionPerformed(ActionEvent e){
                try{
                        jep.setPage("./Aide/Conseil.htm#Statistiques");
                }catch(IOException ioe){System.out.println("Erreur chgt jep");}
                jep.scrollToReference("Statistiques");
        }

        private String chemin(TreePath tp){
                StringTokenizer st = new StringTokenizer(tp.toString(), "[, ]");
                String ficSelect = new String(".");
                while(st.hasMoreElements()){
                        ficSelect = new String(ficSelect +"/"+ st.nextToken() );
                }
                return ficSelect;
        }

        public void valueChanged(TreeSelectionEvent tse){
                try{
                        JTree tree = (JTree)tse.getSource();
                        TreePath tp = tree.getSelectionPath();
                        String dernier = tp.getLastPathComponent().toString();

                        if(tp.getPathCount()==2){

                                arbo.chargerEditor(chemin(tp));
                                        System.out.println("Charger " + chemin(tp)
);
                        }
                        if(tp.getPathCount()==3){
                                arbo.chargerEditor(chemin(tp));
                                jep.scrollToReference(dernier);
                                System.out.print("Charger " + chemin(tp.getParentP
ath()));
                                System.out.println(" // " + dernier);
                        }
                        jep.repaint();
                }catch(NullPointerException npe){};
        }

}
```

```java
import javax.swing.*;
import java.awt.*;

public class DButton extends JButton{

    int x,y;

    public DButton(int i, int j){
        super();
        x = i;
        y = j;
    }

    public int getX(){
        return x;
    }

    public int getY(){
        return y;
    }
}
```

```java
public class DCase{

    private boolean mine;
    private int mimesAlentour;
    private boolean drapeau;
    private boolean decouverte;
    private boolean selection;

    public DCase(){
        mine = false;
        decouverte = false;
        drapeau = false;
        selection = false;
        mimesAlentour = -1;
    }


    public boolean estMine(){
        return mine;
    }

    public int getMinesAlentour(){
        return mimesAlentour;
    }

    public boolean yaDrapreau(){
        return drapeau;
    }

    public boolean estDecouverte(){
        return decouverte;
    }

    public void setDecouverte(){
        decouverte = true;
    }

    public void selectionner(){
        selection = true;
    }

    public void deselectionner(){
        selection = false;
    }

    public boolean select(){
        return selection;
    }

    public void drapeauAction(){
        if(!decouverte)
            drapeau = (!drapeau);

    }

    public void poserBombe(){
        mine = true;
    }

    public void setMinesAlentour(int nb){
        mimesAlentour = nb;
    }
}
```

```java
import java.awt.*;
import javax.swing.*;

public class DChronoLabel extends JLabel implements Runnable{

    private Thread t;
    private boolean enMarche, pause;
    private int temps;

    public DChronoLabel(){
        super();
        initChrono();
    }

    public boolean estActif(){  return enMarche;    }

    public Thread getThread(){  return t;    }


    public void run(){
        temps = 0;
        enMarche = true;
        pause = false;
        while(enMarche){
            temps++;
            setText(convertirTemps());
            while(pause){ }
            try{
                t.sleep(1000);
            }
            catch (InterruptedException ie){;}
        }
    }

    public void mettreEnPause(){ pause = true;    }

    public void enleverPause(){ pause = false;    }

    public void stop(){ enMarche = false;    }

    String convertirTemps(){
        Integer i = new Integer(temps);
        if(temps<10)
            return ("00"+i.toString());
        else
            if(temps<100)
                return ("0"+i.toString());
            else
                return i.toString();
    }

    public void initChrono(){
        setText("000");
        temps = 0;
        enMarche = false;
        t = new Thread(this);
    }

    public int getTime(){return temps;}

}
```

```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class DFenetre extends JFrame {

    final static int DEBUTANT = 1;
    final static int INTER = 2;
    final static int EXPERT = 3;
    final static int PERSO = 4;

    int nb_col;
    int nb_lgn;
    int nb_mines;

    JMenuBar barreMenus;
    JMenu jeu, options, aPropos;

    public JRadioButtonMenuItem debutant, intermediaire, expert, perso;
    public JMenuItem nouvelle, quitter, aide, createur, design, stat;

    private JLabel minesRestantes;
    private DChronoLabel temps;
    private JPanel nord;
    private int tailleX, tailleY;

    private JButton go;
    private DButton[][] boutons;
    private DPartie partie;

    private DImageur imageur;
    private DPanneau centre;

    int type;

    public DFenetre(DPartie p){
        super("Demineur");
        menu();
        type = DEBUTANT;
        imageur = new DImageur(p);
        miseEnPage();
        connecterPartie(p);
    }

    public void connecterPartie(DPartie p){
        partie = p;
        nb_lgn = p.getMatrice().getHauteur();
        nb_col = p.getMatrice().getLargeur();
        nb_mines = p.getMatrice().getMines();

        miseAJourCompteur();
        goCool();

        /* partie centrale : damier */
        if(centre!=null)
            getContentPane().remove(centre);
        centre = new DPanneau(imageur, nb_lgn, nb_col);

        EcouteurSouris ecouteurSouris =
            new EcouteurSouris(this, partie);
        centre.addMouseListener(ecouteurSouris);
        centre.addMouseMotionListener(ecouteurSouris);

        getContentPane().add(centre, BorderLayout.CENTER);

        /* Affichage */
```



```java
        this.setSize(20*nb_col + 6, 20*nb_lgn + 50 + 23 + 25);
        this.setResizable(false);
        this.setVisible(true);
        this.repaint();

    }

    private void menu(){
        /* creation du menu de jeu */
        jeu  = new JMenu("Jeu");
        nouvelle = new JMenuItem("Nouvelle partie");
        nouvelle.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_N,
                                            InputEvent.CTRL_MASK));
        nouvelle.setToolTipText("Partie avec les mêmes paramètres");
        jeu.add(nouvelle);
        jeu.addSeparator();

        ButtonGroup groupRadio=new ButtonGroup();

        debutant = new JRadioButtonMenuItem("Débutant");
        debutant.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_D,
                                            InputEvent.CTRL_MASK));
        debutant.setToolTipText("81 cases 10 mines");

        intermediaire = new JRadioButtonMenuItem("Intermédiaire");
        intermediaire.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_I,
                                            InputEvent.CTRL_MASK));
        intermediaire.setToolTipText("256 cases 40 mines");

        expert = new  JRadioButtonMenuItem("Expert");
        expert.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_E,
                                            InputEvent.CTRL_MASK));
        expert.setToolTipText("480 cases 99 mines");

        perso = new  JRadioButtonMenuItem("Personnalisé...");
        perso.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_P,
                                            InputEvent.CTRL_MASK));
        perso.setToolTipText("Partie avec vos votres paramètres");

        jeu.add(debutant);
        jeu.add(intermediaire);
        jeu.add(expert);
        jeu.add(perso);

        groupRadio.add(debutant);
        groupRadio.add(intermediaire);
        groupRadio.add(expert);
        groupRadio.add(perso);

        jeu.addSeparator();
        quitter = new JMenuItem("Quitter");
        quitter.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_Q,
                                            InputEvent.CTRL_MASK));

        jeu.add(quitter);

        /* creation du menu de options */
        options = new JMenu("Options");

        design = new JMenuItem("Graphisme");
        design.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_G,
                                            InputEvent.CTRL_MASK));
        design.setToolTipText("Pour choisir le style d'image");
        options.add(design);
        options.addSeparator();

        stat = new JMenuItem("Statistiques");
```

```java
        stat.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_S,
                                                   InputEvent.CTRL_MASK));
        stat.setToolTipText("Pour connaitre les scores");
        options.add(stat);

        /* creation du menu A propos */
        aPropos = new JMenu("?");

        aide = new JMenuItem("Aide");
        aide.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_A,
                                                   InputEvent.CTRL_MASK));
        aide.setToolTipText("Pour obtenir de l'aide");
        aPropos.add(aide);

        aPropos.addSeparator();

        createur = new JMenuItem("Créateurs");
        createur.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_C,
                                                       InputEvent.CTRL_MASK));
        createur.setToolTipText("Par qui ?");
        aPropos.add(createur);

        /* ajout des menus à la barre */
        barreMenus = new JMenuBar();
        barreMenus.add(jeu);
        barreMenus.add(options);
        barreMenus.add(aPropos);
        this.setJMenuBar(barreMenus);
    }

    public void ecouterMenu(ActionListener al){
        debutant.addActionListener(al);
        intermediaire.addActionListener(al);
        expert.addActionListener(al);
        perso.addActionListener(al);
        nouvelle.addActionListener(al);
        quitter.addActionListener(al);
        design.addActionListener(al);
        stat.addActionListener(al);
        aide.addActionListener(al);
        createur.addActionListener(al);

    }

    public JMenuItem getNouvelle(){
        return nouvelle;
    }

    public JMenuItem getDebutant(){
        return debutant;
    }

    public JMenuItem getIntermediaire(){
        return intermediaire;
    }

    public JMenuItem getExpert(){
        return expert;
    }

    public JMenuItem getPerso(){
        return perso;
    }

    public JMenuItem getDesign(){
        return design;
    }
```



```java
    public JMenuItem getStat(){
        return stat;
    }

    public JMenuItem getQuitter(){
        return quitter;
    }

    public JMenuItem getAide(){
        return aide;
    }

    public JMenuItem getCreateur(){
        return createur;
    }

    public int getHauteur(){
        return nb_lgn;
    }

    public int getLargeur(){
        return nb_col;
    }

    public int getMines(){
        return nb_mines;
    }

    private void miseEnPage(){
        Container c = this.getContentPane();
        c.setLayout(new BorderLayout(5, 5));

        /* partie haute de l'IHM */
        nord = new JPanel();
        nord.setLayout(new GridBagLayout());
        GridBagConstraints gbc = new GridBagConstraints();
        gbc.anchor = GridBagConstraints.CENTER;
        gbc.weightx = 1;
        gbc.weighty = 1;
        gbc.insets = new Insets(5,0,1,0);

        /* en haut a gauche */
        minesRestantes = new JLabel("00");
        //nord.add(minesRestantes, BorderLayout.WEST);
        nord.add(minesRestantes, gbc);
        /* centre */
        go = new JButton();

        goCool();
        nord.add(go, gbc);

        temps = new DChronoLabel();
        nord.add(temps,gbc);
        c.add(nord, BorderLayout.NORTH);
    }

    public JButton getGo(){
        return go;
    }

    public JMenu getMenuPartie(){
        return jeu;
    }

    public DPanneau getPanneauCentral(){
        return centre;
```

```
    }

    public DImageur getImageur(){
        return imageur;
    }

    public void goPerdu(){
        go.setIcon(new ImageIcon(imageur.getRepertoire()+"/Perdu.GIF"));
    }

    public void goGagne(){
        go.setIcon(new ImageIcon(imageur.getRepertoire()+"/Gagne.GIF"));
    }

    public void goOups(){
        go.setIcon(new ImageIcon(imageur.getRepertoire()+"/Oups.GIF"));
    }

    public void goCool(){
        go.setIcon(new ImageIcon(imageur.getRepertoire()+"/Cool.GIF"));
    }

    public void miseAJourCompteur(){
        int nb = partie.getMatrice().getMines()
            -partie.getMatrice().nbrDrapeau();
        Integer integer = new Integer(nb);
        if((nb>9) || (nb<0))
            minesRestantes.setText(integer.toString());
        else
            minesRestantes.setText("0"+integer.toString());
    }

    public void lancerChrono(){
        if(!temps.estActif())
            temps.getThread().start();
    }
    public void arretChrono(){
        temps.stop();
    }

    public void initChrono(){
        temps.initChrono();
    }

    public void pauseChrono(){
        temps.mettreEnPause();
    }

    public void repriseChrono(){
        temps.enleverPause();
    }

    public int getChrono(){
        return temps.getTime();
    }
}
```

```java
import javax.swing.*;
import java.awt.event.*;
import java.io.*;
import java.util.*;

public class DImageChooser extends JFileChooser {

    public DImageChooser(DImageur imageur, DPanneau panneau){
        super("./Images/");
        setApproveButtonText("OK");
        setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
        int returnVal = this.showOpenDialog(new JFrame());
        if(returnVal == JFileChooser.APPROVE_OPTION){
            String path = getSelectedFile().getPath();
            String fichier;
            if((fichier= estValide(path))==null){
                panneau.repaint();
                imageur.setRepertoire(path);
            }
            else
                JOptionPane.showMessageDialog(this,
                     fichier + " est manquant ou invalide"
                    ,"Répertoire invalide",JOptionPane.ERROR_MESSAGE);
        }
    }


    public String estValide(String rep){
        int i;
        File path = new File(rep);
        String[] fichiers = path.list();
        int nbrFichiers = fichiers.length;


        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("1.GIF"))
                break;
        if(i==nbrFichiers)
            return "1.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("2.GIF"))
                break;
        if(i==nbrFichiers)
            return "2.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("3.GIF"))
                break;
        if(i==nbrFichiers)
            return "3.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("4.GIF"))
                break;
        if(i==nbrFichiers)
            return "4.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("5.GIF"))
                break;
        if(i==nbrFichiers)
            return "5.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("6.GIF"))
                break;
```



```java
        if(i==nbrFichiers)
            return "6.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("7.GIF"))
                break;
        if(i==nbrFichiers)
            return "7.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("8.GIF"))
                break;
        if(i==nbrFichiers)
            return "8.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("Cool.GIF"))
                break;
        if(i==nbrFichiers)
            return "Cool.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("Croix.GIF"))
                break;
        if(i==nbrFichiers)
            return "Croix.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("Drapeau.GIF"))
                break;
        if(i==nbrFichiers)
            return "Drapeau.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("Gagne.GIF"))
                break;
        if(i==nbrFichiers)
            return "Gagne.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("Inconnue.GIF"))
                break;
        if(i==nbrFichiers)
            return "Inconnue.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("Mine.GIF"))
                break;
        if(i==nbrFichiers)
            return "Mine.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("Oups.GIF"))
                break;
        if(i==nbrFichiers)
            return "Oups.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("Perdu.GIF"))
                break;
        if(i==nbrFichiers)
            return "Perdu.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("Select.GIF"))
                break;
        if(i==nbrFichiers)
```

```
            return "Select.GIF";

        for(i=0; i<nbrFichiers ;i++)
            if(fichiers[i].equals("Vide.GIF"))
                break;
        if(i==nbrFichiers)
            return "Vide.GIF";
        return null;
    }
}
```

```java
import javax.swing.*;
import java.awt.*;

public class DImageur{

    private DPartie partie;
    public String repertoire = "./Images/Classic";

    public DImageur(DPartie p){
        partie = p;
    }

    public String getRepertoire(){
        return repertoire;
    }

    public void setRepertoire(String s){
        repertoire = s;
    }


    ImageIcon getImage(int i, int j){
        if(!partie.perdu() && !partie.gagne()){
            if(partie.getMatrice().getCase(i,j).yaDrapreau())
                return new ImageIcon(repertoire+"/Drapeau.GIF");
            if(!partie.getMatrice().getCase(i,j).estDecouverte()){
                if(partie.getMatrice().getCase(i,j).select())
                    return new ImageIcon(repertoire+"/Select.GIF");
                else
                    return new ImageIcon(repertoire+"/Inconnue.GIF");
            }
            switch(partie.getMatrice().getCase(i,j).getMinesAlentour()){
            case 0: return new ImageIcon(repertoire+"/Vide.GIF");
            case 1: return new ImageIcon(repertoire+"/1.GIF");
            case 2: return new ImageIcon(repertoire+"/2.GIF");
            case 3: return new ImageIcon(repertoire+"/3.GIF");
            case 4: return new ImageIcon(repertoire+"/4.GIF");
            case 5: return new ImageIcon(repertoire+"/5.GIF");
            case 6: return new ImageIcon(repertoire+"/6.GIF");
            case 7: return new ImageIcon(repertoire+"/7.GIF");
            case 8: return new ImageIcon(repertoire+"/8.GIF");
            default : return new ImageIcon(repertoire+"/Mine.GIF");
            }
        }
        else{
            if(partie.perdu()){
                if((partie.getMatrice().getCase(i,j).yaDrapreau())
                   && !(partie.getMatrice().getCase(i,j).estMine() ))
                    return new ImageIcon(repertoire+"/Croix.GIF");
                if(partie.getMatrice().getCase(i,j).estMine())
                    return new ImageIcon(repertoire+"/Mine.GIF");
                if(!partie.getMatrice().getCase(i,j).estDecouverte())
                    return new ImageIcon(repertoire+"/Inconnue.GIF");
                switch(partie.getMatrice().getCase(i,j).getMinesAlentour()){
                case 0: return new ImageIcon(repertoire+"/Vide.GIF");
                case 1: return new ImageIcon(repertoire+"/1.GIF");
                case 2: return new ImageIcon(repertoire+"/2.GIF");
                case 3: return new ImageIcon(repertoire+"/3.GIF");
                case 4: return new ImageIcon(repertoire+"/4.GIF");
                case 5: return new ImageIcon(repertoire+"/5.GIF");
                case 6: return new ImageIcon(repertoire+"/6.GIF");
                case 7: return new ImageIcon(repertoire+"/7.GIF");
                case 8: return new ImageIcon(repertoire+"/8.GIF");
                default : return new ImageIcon(repertoire+"/Mine.GIF");
                }
            }
            else {
```



```java
                switch(partie.getMatrice().getCase(i,j).getMinesAlentour()){
                case 0: return new ImageIcon(repertoire+"/Vide.GIF");
                case 1: return new ImageIcon(repertoire+"/1.GIF");
                case 2: return new ImageIcon(repertoire+"/2.GIF");
                case 3: return new ImageIcon(repertoire+"/3.GIF");
                case 4: return new ImageIcon(repertoire+"/4.GIF");
                case 5: return new ImageIcon(repertoire+"/5.GIF");
                case 6: return new ImageIcon(repertoire+"/6.GIF");
                case 7: return new ImageIcon(repertoire+"/7.GIF");
                case 8: return new ImageIcon(repertoire+"/8.GIF");
                default : return new ImageIcon(repertoire+"/Drapeau.GIF");
                }
            }
        }
    }

}
```

```java
import java.lang.*;
import java.util.Random;

public class DMatrice{

    private int hauteur, largeur, nbMines; // parametres de la partie
    private DCase[][] matrice;
    private int caseNonMineeRestante;
    private boolean explosion;

    public DMatrice(int h, int l, int nb){
        hauteur = h;
        largeur = l;
        nbMines = nb;
        explosion = false;
        matrice = new DCase[h][l];
        for(int i=0; i<hauteur; i++)
            for(int j=0; j<largeur; j++)
                matrice[i][j] = new DCase();
        miner();
        preparerAlentour();
        caseNonMineeRestante = hauteur*largeur-nbMines;
    }

    public DCase getCase(int i, int j)  {
        try{
            return matrice[i][j];
        }
        catch(ArrayIndexOutOfBoundsException e){ return null; }
    }

    public int getHauteur(){
        return hauteur;
    }

    public int getLargeur(){
        return largeur;
    }

    public int getMines(){
        return nbMines;
    }


    public void devoilerCase(int i,int j){
        /* Case découverte */
        try{
            matrice[i][j].setDecouverte();
            caseNonMineeRestante--;
        }
        catch(ArrayIndexOutOfBoundsException e){   }


        /* on regarde si la case est minée */
        try{
            if(matrice[i][j].estMine())
                explosion = true;
            else{
                /* propagation éventuelle */
                if(matrice[i][j].getMinesAlentour()==0){
                    try{
                        if(!matrice[i-1][j-1].estDecouverte())
                            devoilerCase(i-1,j-1);
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
```



```java
                        if(!matrice[i-1][j].estDecouverte())
                            devoilerCase(i-1,j);
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
                        if(!matrice[i-1][j+1].estDecouverte())
                            devoilerCase(i-1,j+1);
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
                        if(!matrice[i][j-1].estDecouverte())
                            devoilerCase(i,j-1);
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
                        if(!matrice[i][j+1].estDecouverte())
                            devoilerCase(i,j+1);
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
                        if(!matrice[i+1][j-1].estDecouverte())
                            devoilerCase(i+1,j-1);
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
                        if(!matrice[i+1][j].estDecouverte())
                            devoilerCase(i+1,j);
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
                        if(!matrice[i+1][j+1].estDecouverte())
                            devoilerCase(i+1,j+1);
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }
                }
            }
        }
        catch(ArrayIndexOutOfBoundsException e){   }
    }

    public void drapeauAction(int i, int j){
        matrice[i][j].drapeauAction();
    }

    private void preparerAlentour(){
        int minesCompteur;

        for(int i=0; i<hauteur; i++)
            for(int j=0; j<largeur; j++){
                minesCompteur=0;
                if(!matrice[i][j].estMine()){
                    try{
                        if(matrice[i-1][j-1].estMine())
                            minesCompteur++;
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
                        if(matrice[i-1][j].estMine())
                            minesCompteur++;
                    }
```

```
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
                        if(matrice[i-1][j+1].estMine())
                            minesCompteur++;
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
                        if(matrice[i][j-1].estMine())
                            minesCompteur++;
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
                        if(matrice[i][j+1].estMine())
                            minesCompteur++;
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
                        if(matrice[i+1][j-1].estMine())
                            minesCompteur++;
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
                        if(matrice[i+1][j].estMine())
                            minesCompteur++;
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    try{
                        if(matrice[i+1][j+1].estMine())
                            minesCompteur++;
                    }
                    catch(ArrayIndexOutOfBoundsException e){   }

                    /* les mines ont étés comptés*/
                    matrice[i][j].setMinesAlentour(minesCompteur);
                }
            }
    }


    private void miner(){
        int x,y;
        int i=0;
        Random alea = new Random();
        while(i<nbMines){
            x = alea.nextInt(hauteur);
            y = alea.nextInt(largeur);
            if(!matrice[x][y].estMine()){
                matrice[x][y].poserBombe();
                i++;
            }

        }
    }

    public int nbrDrapeau(){
        int compteur = 0;
        for(int i=0;i<hauteur;i++)
            for(int j=0;j<largeur;j++){
                if(matrice[i][j].yaDrapeau())
                    compteur++;
            }
```



```
        return compteur;
    }

    public boolean aExplose(){
        return explosion;
    }

    public int getCaseNonMineeRestante(){
        return caseNonMineeRestante;
    }

    public void afficher(){
        System.out.println("****carte****");
        for(int i=0; i<hauteur; i++){
            for(int j=0; j<largeur; j++){
                if(matrice[i][j].estMine())
                    System.out.print("M ");
                else
                    System.out.print(matrice[i][j].getMinesAlentour()+" ");
            }
            System.out.println("");

        }
        System.out.println("****carte connue****");
        for(int i=0; i<hauteur; i++){
            for(int j=0; j<largeur; j++){
                if(!matrice[i][j].estDecouverte())
                    System.out.print("* ");
                else
                    if(matrice[i][j].estMine())
                        System.out.print("M ");
                    else
                        System.out.print(matrice[i][j].getMinesAlentour()+" ");
            }
            System.out.println("");
        }
    }
}
```

```java
import javax.swing.*;
import java.awt.*;

public class DPanneau extends JPanel{

    private DImageur imageur;
    private int hauteur, largeur;

    public DPanneau(DImageur im, int h, int l){
        super();
        imageur = im;
        hauteur = h;
        largeur = l;
    }

    public void paintComponent(Graphics g){
        super.paintComponent(g);
        for(int i=0; i<hauteur; i++)
            for(int j=0;j<largeur; j++){
                g.drawImage(imageur.getImage(i,j).getImage(),j*20,i*20,this);
            }
    }
}
```

```
public class DPartie {

    private DMatrice dm;
    private boolean commence, perdu;

    public DPartie(int h, int l, int nb){
        nouvellePartie(h,l,nb);
    }

    public void nouvellePartie(int h, int l, int nb){
        commence = false;
        perdu = false;
        dm = new DMatrice(h,l,nb);
    }

    public DMatrice getMatrice(){
        return dm;
    }

    public void commencer(){
        commence = true;
    }

    public boolean gagne(){
        return (dm.getCaseNonMineeRestante()==0);
    }

    public boolean perdu(){
        return dm.aExplose();
    }

    public boolean fini(){
        return(commence && (!perdu()) && (!gagne()));
    }
}
```

```
import java.awt.event.*;

public class EcouteurFenetre extends WindowAdapter{

    private DFenetre fenetre;

    public EcouteurFenetre(DFenetre f){
        fenetre = f;
    }

    public void windowClosing(WindowEvent e){
        System.exit(0);
    }

    public void windowIconified(WindowEvent e){
        fenetre.pauseChrono();
    }

    public void windowDeiconified(WindowEvent e){
        fenetre.repriseChrono();
    }
}
```

```java
import java.awt.event.*;

public class EcouteurGo implements ActionListener{

    DFenetre fenetre;
    DPartie partie;

    public EcouteurGo(DFenetre f, DPartie p){
        fenetre = f;
        partie = p;
    }


    public void actionPerformed(ActionEvent ae){
        fenetre.arretChrono();
        fenetre.initChrono();

        partie.nouvellePartie(partie.getMatrice().getHauteur(),
                              partie.getMatrice().getLargeur(),
                              partie.getMatrice().getMines());
        fenetre.connecterPartie(partie);
    }
}
```

```
import java.awt.event.*;
import javax.swing.*;
import java.io.*;

public class EcouteurMenu implements ActionListener{

    private DPartie partie;
    private DFenetre fenetre;

    public EcouteurMenu(DFenetre f, DPartie p){
        partie = p;
        fenetre = f;
    }


    public void actionPerformed(ActionEvent ae){
        if(ae.getSource() == fenetre.getNouvelle()){
            fenetre.arretChrono();
            fenetre.initChrono();
            partie.nouvellePartie(partie.getMatrice().getHauteur(),
                                  partie.getMatrice().getLargeur(),
                                  partie.getMatrice().getMines());
            fenetre.connecterPartie(partie);
        }
        if(ae.getSource() == fenetre.getDebutant()){
            fenetre.arretChrono();
            fenetre.initChrono();
            partie.nouvellePartie(9,9,10);
            fenetre.type = fenetre.DEBUTANT;
            fenetre.connecterPartie(partie);
        }

        if(ae.getSource() == fenetre.getIntermediaire()){
            fenetre.arretChrono();
            fenetre.initChrono();
            partie.nouvellePartie(16,16,40);
            fenetre.type = fenetre.INTER;
            fenetre.connecterPartie(partie);
        }
        if(ae.getSource() == fenetre.getExpert()){
            fenetre.arretChrono();
            fenetre.initChrono();
            partie.nouvellePartie(16,30,99);
            fenetre.type = fenetre.EXPERT;
            fenetre.connecterPartie(partie);
        }

        if(ae.getSource() == fenetre.getPerso()){
            Pref pref = new Pref(fenetre, partie);
            fenetre.type = fenetre.PERSO;
        }

        if(ae.getSource() == fenetre.getDesign()){
            DImageChooser choix =
                new DImageChooser(fenetre.getImageur(),
                                  fenetre.getPanneauCentral() );
        }

        if(ae.getSource() == fenetre.getQuitter())
            System.exit(0);

        if(ae.getSource()==fenetre.getAide()){
            File f = new File("./Aide");
            Aide a = new Aide(f);;
        }
        if(ae.getSource()==fenetre.getCreateur())
            JOptionPane.showMessageDialog(fenetre,
```



```
                "Réalisé par Igor DAURIAC et Nicolas FRANCOIS,  Projet IHM",
                "Créateurs...",JOptionPane.INFORMATION_MESSAGE);
    }
}
```

```
import java.awt.event.*;
import java.io.*;


public class EcouteurSouris  implements MouseListener, MouseMotionListener{

    private DPartie partie;
    private DFenetre fenetre;
    private int sourisX, sourisY;
    private boolean gauchePresse;

    public EcouteurSouris(DFenetre f, DPartie p){
        partie = p;
        fenetre = f;
        gauchePresse = false;
    }

    public void mouseReleased(MouseEvent me){
        sourisX = me.getX()/20;
        sourisY = me.getY()/20;
        try{
            if(!partie.perdu() && !partie.gagne()
               && !(partie.getMatrice().getCase(sourisY,
                                            sourisX).yaDrapreau())){

                if(me.getButton()==me.BUTTON1){
                    gauchePresse = false;
                    try{
                        partie.getMatrice().getCase(sourisY,
                                               sourisX).selectionner();
                    }
                    catch(NullPointerException npe){  }
                    try{
                        partie.getMatrice().devoilerCase(sourisY,sourisX);
                    }
                    catch(NullPointerException npe){  }
                    fenetre.lancerChrono();
                    if(partie.gagne()){
                        fenetre.goGagne();
                        fenetre.arretChrono();
                    }
                    else
                        if(partie.perdu()){
                            fenetre.goPerdu();
                            fenetre.arretChrono();
                        }
                        else
                            fenetre.goCool();
                }
            }
        }
        catch(NullPointerException npe){  }
        me.getComponent().repaint();
    }

    public void mousePressed(MouseEvent me){
        sourisX = me.getX()/20;
        sourisY = me.getY()/20;
        if(!partie.perdu() && !partie.gagne()){
            if(me.getButton()==me.BUTTON1){
                gauchePresse = true;
                partie.getMatrice().getCase(sourisY,sourisX).selectionner();
                fenetre.goOups();
            }
            if(me.getButton()==me.BUTTON3){
                partie.getMatrice().drapeauAction(sourisY,sourisX);
                fenetre.miseAJourCompteur();
```



```
            }
        }
        me.getComponent().repaint();
    }

    public void mouseExited(MouseEvent me){
        if(gauchePresse)
            fenetre.goCool();
    }

    public void mouseEntered(MouseEvent me){
        if(gauchePresse)
            fenetre.goOups();
    }

    public void mouseDragged(MouseEvent me){
        int x = me.getX()/20;
        int y = me.getY()/20;
        if(((x!=sourisX) || (y!=sourisY)) && gauchePresse){
            try{
                partie.getMatrice().getCase(sourisY,
                                          sourisX).deselectionner();
            }
            catch(NullPointerException npe){  }
            sourisX = x;
            sourisY = y;
            try{
                partie.getMatrice().getCase(sourisY,
                                          sourisX).selectionner();
            }
            catch(NullPointerException npe){  }
            me.getComponent().repaint();
        }
    }
    public void mouseMoved(MouseEvent me){    }

    public void mouseClicked(MouseEvent e){    }
}
```

```
import java.io.*;
import java.util.*;

public class Main{
    public static void main(String[] args){
        DPartie partie = new DPartie(9,9,10);
        DFenetre fenetre = new DFenetre(partie);

        EcouteurGo e = new EcouteurGo(fenetre,partie);
        fenetre.getGo().addActionListener(e);

        EcouteurFenetre ecouteurFenetre = new EcouteurFenetre(fenetre);
        fenetre.addWindowListener(ecouteurFenetre);

        EcouteurMenu ecouteurMenu = new EcouteurMenu(fenetre,partie);
        fenetre.ecouterMenu(ecouteurMenu);
    }
}
```

```
import javax.swing.*;
import java.awt.*;
import java.io.*;

public class Onglet{
    JFrame maFen;
    JTabbedPane jtp;

    private JPanel panelCarte(){
        JPanel panel = new JPanel();
        panel.setLayout(new BorderLayout(10,10));
        panel.add(new JTextArea("Carte"), BorderLayout.CENTER);
        return panel;
    }

    private JPanel panelCasierJud(){
        JPanel panel = new JPanel();
        panel.setLayout(new BorderLayout(10,10));
        panel.add(new JTextArea("Casier Judiciaire"),
                  BorderLayout.CENTER);
        return panel;
    }

    private JPanel panelDebutant(){
        JPanel panel = new JPanel();
        panel.setLayout(new BorderLayout(10,10));
        panel.add(new JTextArea("Debutant"), BorderLayout.CENTER);
        return panel;
    }

    private JPanel panelConfirme(){
        JPanel panel = new JPanel();
        panel.setLayout(new BorderLayout(10,10));
        panel.add(new JTextArea("Confirme"), BorderLayout.CENTER);
        return panel;
    }

    private JPanel panelExpert(){
        JPanel panel = new JPanel();
        panel.setLayout(new BorderLayout(10,10));
        panel.add(new JTextArea("Expert"), BorderLayout.CENTER);
        return panel;
    }

    private void miseEnPage(){
        Container c = maFen.getContentPane();
        c.setLayout(new BorderLayout());

        JTabbedPane jtp1 = new JTabbedPane();
        JTabbedPane jtp2 = new JTabbedPane();

        try{
            jtp = new JTabbedPane();
        }catch(IllegalArgumentException iae){
            System.out.println("Erreur construction jtp");
        }
        jtp1.add(panelCarte(), "Carte");
        jtp1.add(panelCasierJud(), "Casier Judiciaire");

        jtp2.add(panelDebutant(), "Debutant");
        jtp2.add(panelConfirme(), "Confirme");
        jtp2.add(panelExpert(), "Expert");

        jtp.add(jtp1, "jtp1");
        jtp.add(jtp2, "jtp2");

        c.add(jtp, BorderLayout.CENTER);
```



```
    }

    public Onglet(){
        maFen = new JFrame("Onglet");
        /* mise en page */
        miseEnPage();
        /* affichage */
        maFen.setSize(300, 200);
        maFen.setVisible(true);
    }
}
```

```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Pref implements ActionListener, FocusListener{

    JFrame maFen;
    JTextField jtfLargeur, jtfHauteur, jtfMines;
    JLabel jlLargeur, jlHauteur, jlMines;
    JButton btnOk, btnAnnuler;

    static int NB_MINES_MIN = 10;
    static int NB_MINES_MAX = 665;
    static int DIMENSION_MIN = 9;
    static int DIMENSION_MAX = 25;

    int hauteur, largeur, mines;

    boolean validation = false;

    private DFenetre fenetre;
    private DPartie partie;

    public Pref(int lng, int lrg, int nbMines){
        maFen = new JFrame("Preferences");
        hauteur = lng;
        largeur = lrg;
        mines = nbMines;
        miseEnPage();
        ajoutEcouteurs();
        maFen.setSize(300, 200);
        maFen.setVisible(true);
    }


    public Pref(DFenetre fen, DPartie p){
        maFen = new JFrame("Preferences");
        fenetre = fen;
        partie = p;
        System.out.println("Choix");

        hauteur = fenetre.getHauteur();
        largeur = fenetre.getLargeur();
        mines = fenetre.getMines();

        miseEnPage();
        ajoutEcouteurs();

        maFen.setSize(300, 200);
        maFen.setVisible(true);
    }


    private JPanel label(){
        /* Creation */
        jlLargeur = new JLabel("Largeur");
        jlHauteur = new JLabel("Hauteur");
        jlMines = new JLabel("Mines");
        JPanel panel = new JPanel();
        /* Ajout */
        panel.setLayout(new GridLayout(3,1, 15,5));
        panel.add(jlLargeur);
        panel.add(jlHauteur);
        panel.add(jlMines);
        /* Retour */
        return panel;
    }
```



```java
    private JPanel saisie(){
        /* Creation */
        jtfLargeur = new JTextField(Integer.toString(largeur));
        jtfHauteur = new JTextField(Integer.toString(hauteur));
        jtfMines = new JTextField(Integer.toString(mines));
        JPanel panel = new JPanel();
        /* Ajout */
        panel.setLayout(new GridLayout(3,1, 15,5));
        panel.add(jtfLargeur);
        panel.add(jtfHauteur);
        panel.add(jtfMines);
        /* Retour */
        return panel;
    }

    private JPanel boutons(){
        /* Creation */
        btnOk = new JButton("Ok");
        btnAnnuler = new JButton("Annuler");
        /* Ajout */
        JPanel panel = new JPanel();
        panel.setLayout(new BorderLayout());
        panel.add(btnOk, BorderLayout.NORTH);
        panel.add(btnAnnuler, BorderLayout.SOUTH);
        /* Retour */
        return panel;
    }

    private void ajoutEcouteurs(){
        btnOk.addActionListener(this);
        btnAnnuler.addActionListener(this);

        jtfLargeur.addFocusListener(this);
        jtfHauteur.addFocusListener(this);
        jtfMines.addFocusListener(this);
    }

    private void miseEnPage(){
        Container c = maFen.getContentPane();

        //  Ajout des lables, zones de saisie et des boutons
        Container container = new Container();
        container.setLayout(new BorderLayout(20, 0));
        container.add(label(), BorderLayout.WEST);
        container.add(saisie(), BorderLayout.CENTER);
        container.add(boutons(), BorderLayout.EAST);

        c.setLayout(new BorderLayout());
        c.add(container, BorderLayout.NORTH);
    }



    public void actionPerformed(ActionEvent e){

        if(e.getSource().equals(btnOk)){
            validation = true;
            maFen.setVisible(false);

            hauteur = Integer.parseInt(jtfHauteur.getText());
            largeur = Integer.parseInt(jtfLargeur.getText());
            mines = Integer.parseInt(jtfMines.getText());
            if (mines>hauteur*largeur)
                mines = hauteur*largeur-5;

            fenetre.arretChrono();
```

```
            fenetre.initChrono();

            partie.nouvellePartie(hauteur,largeur,mines);
            fenetre.connecterPartie(partie);

        }
        else{
            validation = false;
        }
    }

    public void focusGained(FocusEvent e){ }

    public void focusLost(FocusEvent e){
        JTextField jtf = (JTextField)e.getSource();
        int val = Integer.parseInt(jtf.getText());

        if(jtf.equals(jtfMines)){
            if(val<NB_MINES_MIN)
                jtf.setText(Integer.toString(NB_MINES_MIN));
            if(val>NB_MINES_MAX)
                jtf.setText(Integer.toString(NB_MINES_MAX));
        }
        else{
            if(val<DIMENSION_MIN)
                jtf.setText(Integer.toString(DIMENSION_MIN));
            if(val>DIMENSION_MAX)
                jtf.setText(Integer.toString(DIMENSION_MAX));
        }

    }

    public boolean getValidation(){return validation;}

    public int getLargeur(){return largeur; }

    public int getHauteur(){return hauteur;}

    public int getMines(){return mines;}

    public static void main(String[] args){
        Pref p = new Pref(10, 10 ,10);
        System.out.println("Fin");
        System.out.println(p.getLargeur()+" "+p.getHauteur()+" "
                           +p.getMines());
    }
}
```