Programming – best practice

Développer un programme en informatique n'est pas une science exacte. Il n'existe rarement qu'une seule manière ni forcément « la meilleure manière » d'écrire les instructions qui accomplissent la tâche visée.

Au lieu de se limiter au « tant que ça marche, c'est bon », nombreux programmeurs préconisent des consignes qui rendent un programme plus cohérent ou plus lisible :

- ne pas écrire des lignes de codes inutiles
- éviter les redondances, ne pas répéter inutilement des lignes de code
- ne pas effectuer à plusieurs reprises le même calcul
- ne pas mélanger les conditions qui sont indépendantes les unes des autres
- etc.

Illustrations à l'aide d'exemples concrets :

au lieu d'écrire	écrire plutôt
`int a=0;` `a = 2;`	`int a=2;`

affectation inutile

au lieu d'écrire	écrire plutôt
`if (a==0) b=2;` `else if (a!=0) c=2;`	`if (a==0) b=2;` `else c=2;`

condition inutile

au lieu d'écrire	écrire plutôt
`if (found==false) error=404;` `else if (forbidden==true) error=405;`	`if (!found) error=404;` `else if (forbidden) error=405;`

conditions inutiles

au lieu d'écrire	écrire plutôt

`if (a==0) {`
`  b=2;`
`  c=0;`
`} else {`
`  b=3;`
`  c=0;`
`}`
	`if (a==0)`
	`  b=2;`
	`else`
	`  b=3;`
	`c=0;`

instruction commune dans une structure alternative

au lieu d'écrire	écrire plutôt

`if (finished) {`
`  g.setColor(color);`
`  g.fillRect(x,y,w,h);`
`  g.setColor(Color.BLACK);`
`  g.drawRect(x,y,w,h);`
`} else {`
`  g.setColor(Color.BLACK);`
`  g.drawRect(x,y,w,h);`
`}`
	`if (finished) {`
	`  g.setColor(color);`
	`  g.fillRect(x,y,w,h);`
	`}`
	`g.setColor(Color.BLACK);`
	`g.drawRect(x,y,w,h);`

instruction commune dans une structure alternative

au lieu d'écrire	écrire plutôt

`for (int i=0; i<alTrucs.size();i++) {`
`  **g.setCOlor(Color.BLACK);**`
`  g.fillOval(x-5,y-5,10,10);`
`}`
	`g.setCOlor(Color.BLACK);`
	`for (int i=0; i<alTrucs.size();i++)`
	`  g.fillOval(x-5,y-5,10,10);`

répétition inutile

au lieu d'écrire	écrire plutôt

`if (age>60)`
`  if (remise)`
`  prix = km*pricePerKm*0.8;`
`  else`
`  prix = km*pricePerKm*0.9;`
`else`
`  if (remise)`
`  prix = km*pricePerKm*0.9;`
`  else`
`  prix = km*pricePerKm;`
	`double factor=1;`
	`if (age>60) factor -= 0.1;`
	`if (remise) factor -= 0.1;`
	`prix = km*pricePerKm*factor;`

mélange de conditions indépendantes

au lieu d'écrire	écrire plutôt

`if (x>w)`
`  if (y>h) {`
`  dx=-dx;`
`  dy=-dy;`
`  } else {`
`  dx=-dx;`
`  y=y+dy;`
`  }`
`} else {`
`  if (y>h) {`
`  x=x+dx;`
`  dy=-dy;`
`  } else {`
`  x=x+dx;`
`  y=y+dy;`
`  }`
`}`
	`if (x>w)`
	`  dx=-dx;`
	`else`
	`  x=x+dx;`
	
	`if (y>h)`
	`  dy=-dy;`
	`else`
	`  y=y+dy;`

mélange de conditions indépendantes

au lieu d'écrire	écrire plutôt

`g.fillOval(x1-(int) Math.sqrt(Math.pow(x1-x2,2)+Math.pow(y1-y2,2)), y1-(int) Math.sqrt(Math.pow(x1-x2,2)+Math.pow(y1-y2,2)), 2*(int) Math.sqrt(Math.pow(x1-x2,2)+Math.pow(y1-y2,2)), 2*(int) Math.sqrt(Math.pow(x1-x2,2)+Math.pow(y1-y2,2)));`	`radius=(int) Math.sqrt(Math.pow(x1-x2,2)+`
	`  Math.pow(y1-y2,2));`
	
	`g.fillOval(x1-radius, y1-radius,`
	`  2*radius, 2*radius)`

répétition du même calcul

au lieu d'écrire	écrire plutôt

`g.setColor(color);`
`g.fillRect( Math.min(from.x,to.x),`
`  Math.min(from.y,to.y),`
`  Math.abs(from.x-to.x),`
`  Math.abs(from.y,to.y));`
`g.setColor(Color.BLACK);`
`g.drawRect( Math.min(from.x,to.x),`
`  Math.min(from.y,to.y),`
`  Math.abs(from.x-to.x),`
`  Math.abs(from.y,to.y));`
	`x = Math.min(from.x,to.x);`
	`y = Math.min(from.y,to.y);`
	`w = Math.abs(from.x-to.x);`
	`h = Math.abs(from.y,to.y);`
	`g.setColor(color);`
	`g.fillRect(x, y, w, h);`
	`g.setColor(Color.BLACK);`
	`g.drawRect(x, y, w, h);`

répétition du même calcul

au lieu d'écrire	écrire plutôt

`public boolean isIn(Point p) {`
`  radius=(int) Math.sqrt(`
`  Math.pow(from.x-to.x,2)+`
`  Math.pow(from.y-to.y,2));`
`  dist =(int) Math.sqrt(`
`  Math.pow(from.x-p.x,2)+`
`  Math.pow(from.y-p.y,2));`
`  return radius>=dist;`
`}`
	`public boolean isIn(Point p) {`
	`  return distance(from, to) >=`
	`  distance(from, p);`
	`}`
	`public int distance(Point p1, Point p2) {`
	`  return =(int) Math.sqrt(`
	`  Math.pow(p1.x-p2.x,2)+`
	`  Math.pow(p1.y-p2.y,2));`
	`}`

répétition d'instruction semblables, rend le programme moins lisible