





ISTANBUL, TURKEY

LOCATION: KOÇ UNIVERSITY

Dates: June 24th – Thursday July 4th



PRAGUE, CZECH REPUBLIC

LOCATION: CZECH TECHNICAL UNIVERSITY

Dates: Tuesday July 9th – Friday July 19th



BOGOTA, COLOMBIA,

LOCATION: UNIVERSIDAD DE LOS ANDES

Dates: Tuesday June 25th – Friday July 5th



KOUMBIA, GUINEE

LOCATION: LYCEE DE KOUMBIA

Dates: Juillet le 3 – Juillet le 17



LES PROFESSEURS



Zak



Harouna (Max)



Aminata



Mariam (Lisa)



Abigael



Instagram

Coding?



facebook



La banque
d'un monde
qui change



Le coding vous apprend à penser par vous-même



Stanford?



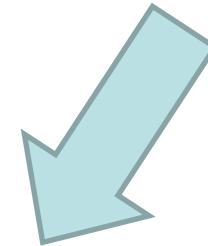
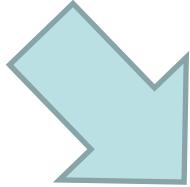
Stanford



Prérequis



Site du Cours

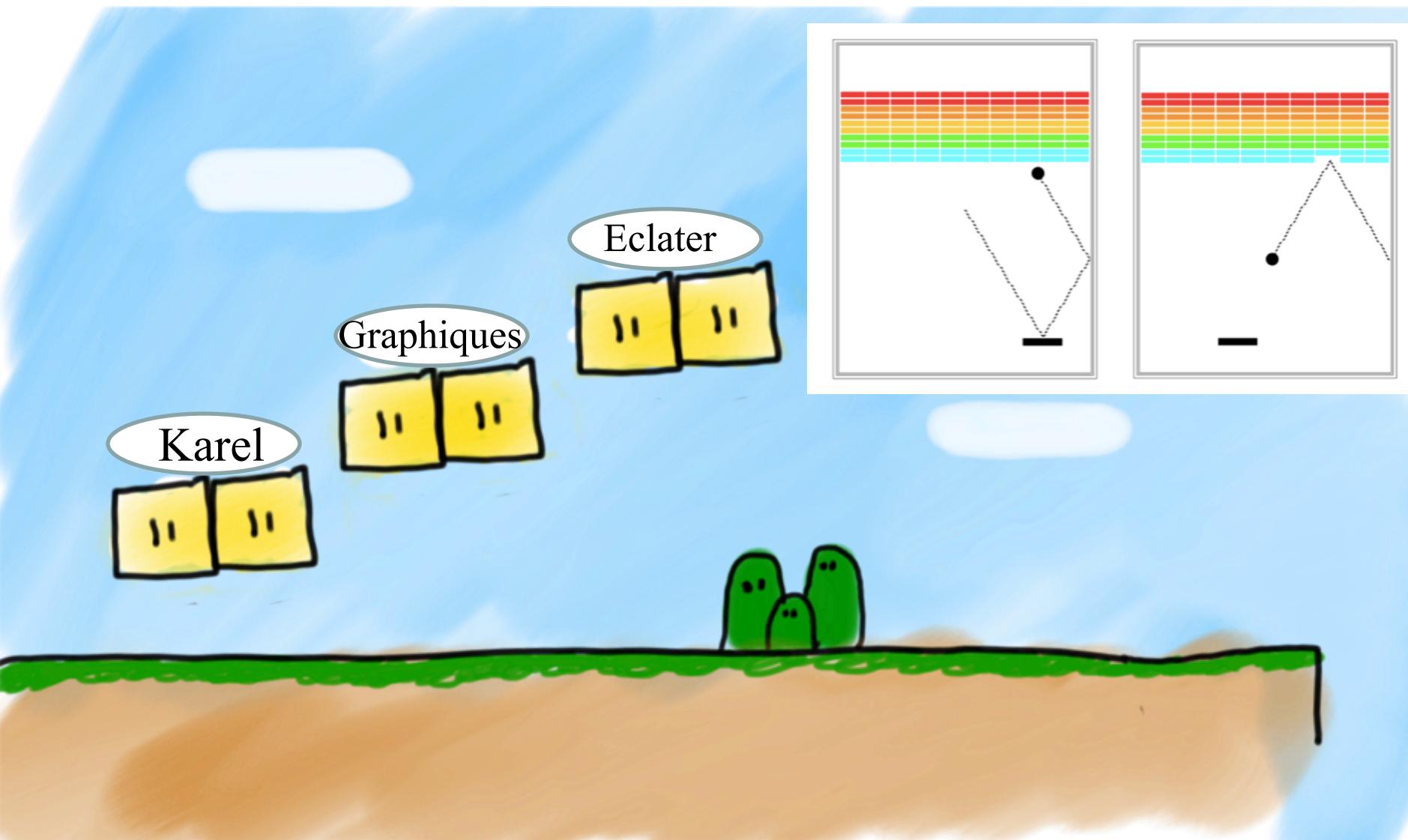


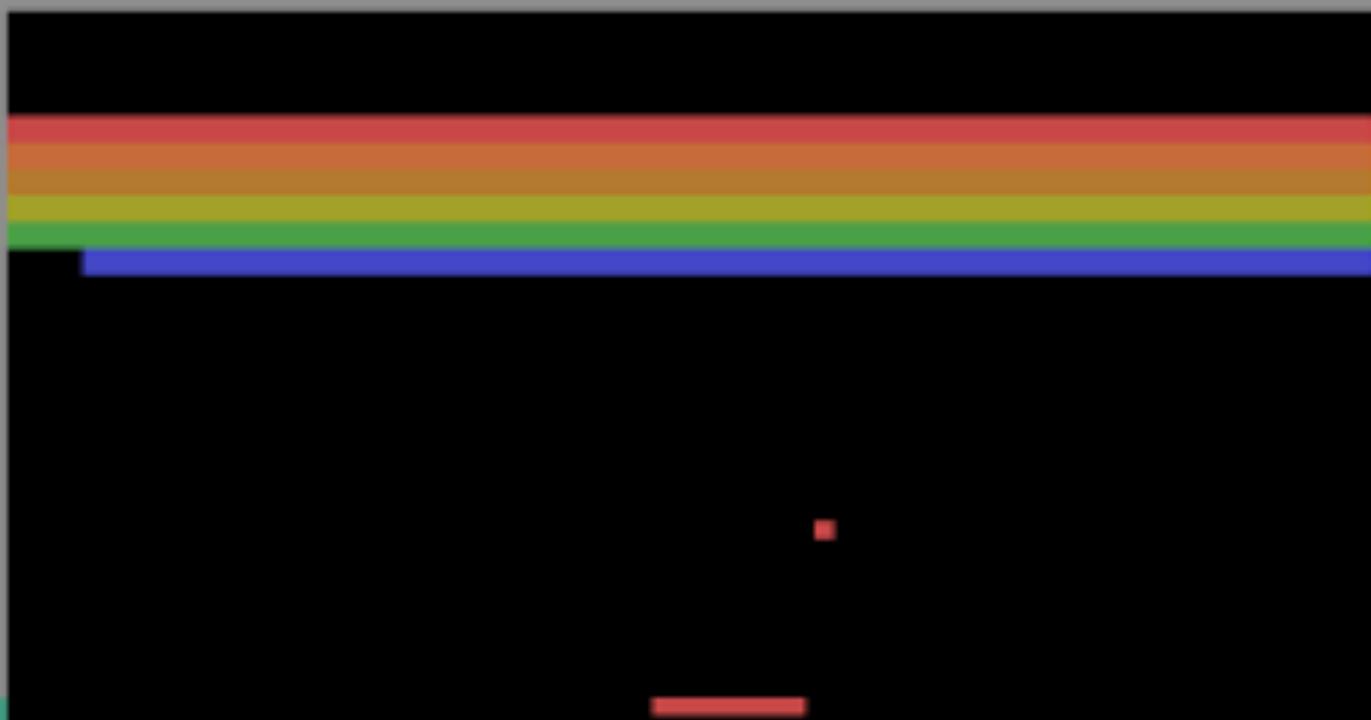
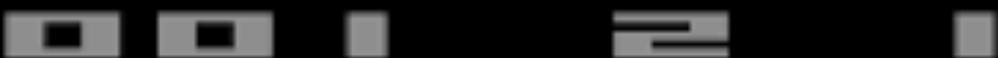
<http://guinea.csbridge.org>



Mais on peut faire sans connection

Le Programme



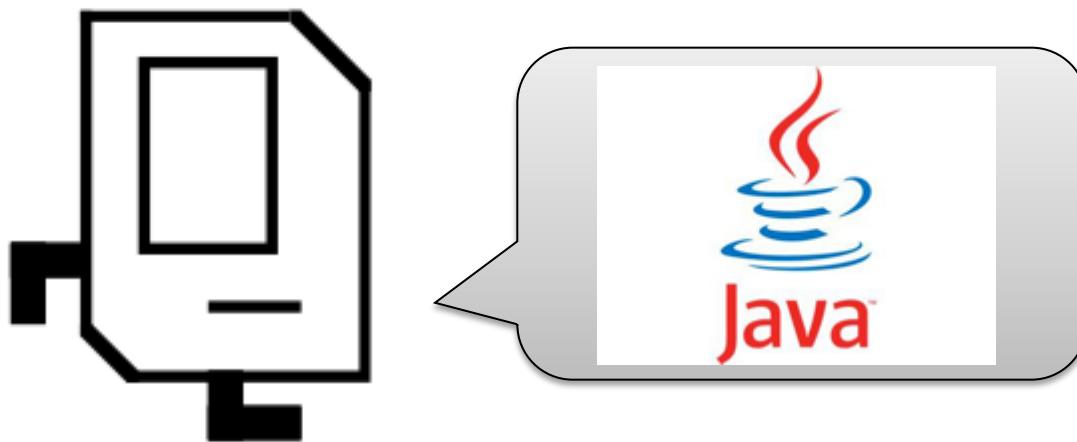


Eclater

Si je le trouve difficile?

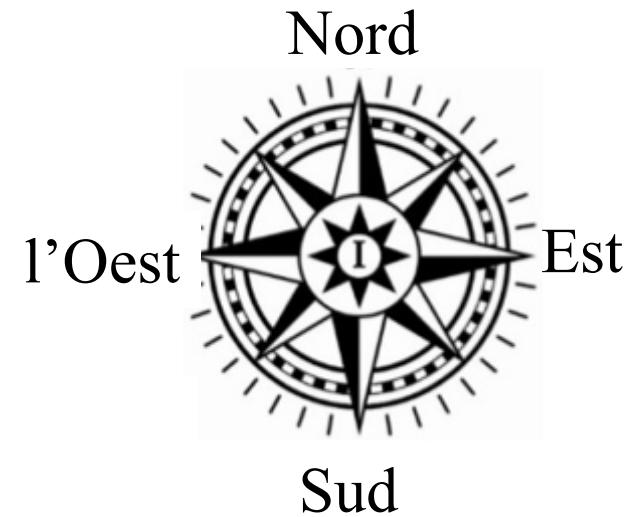
Poser des questions!

Karel parle Java



Le monde de Karel

3	+	+	+	+	
2	+	+	+	+	
1		+	+	+	
	1	2	3	4	5



Elle connaît quatre commandes:

move(); bougez

turnLeft(); tournez à gauche

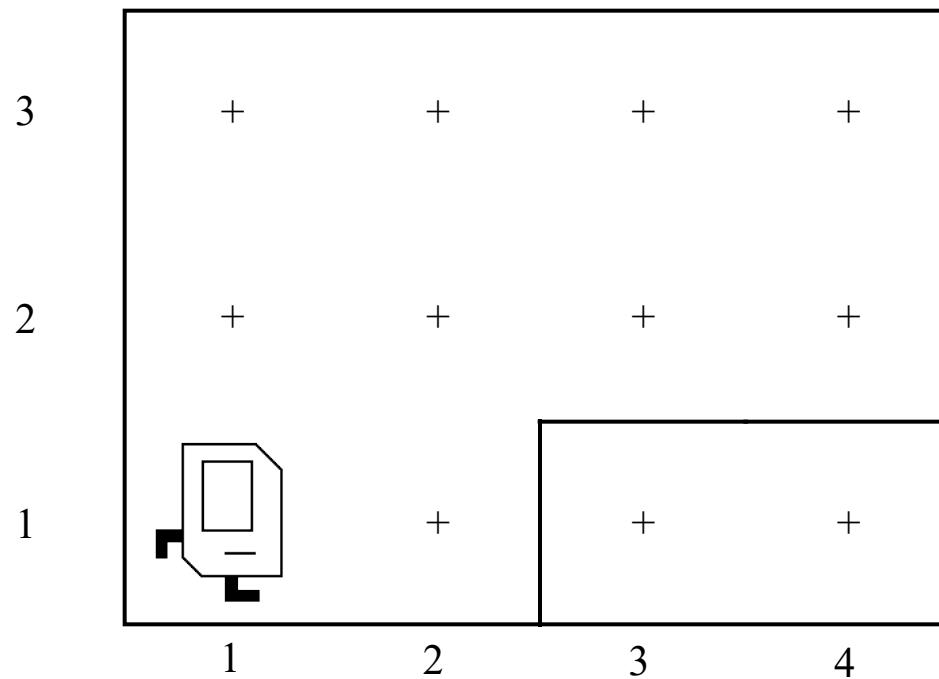
putBeeper(); mettre le bip



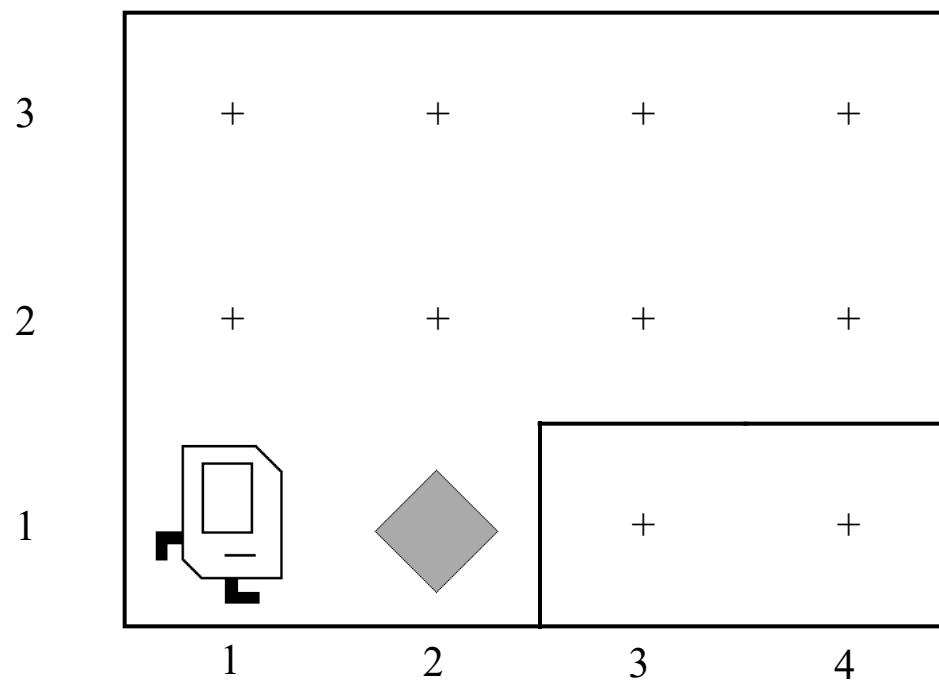
pickBeeper();
prendre le bip

Quelque commandes pour des
ordinateurs sont en anglais...patience

des murs



Des bips

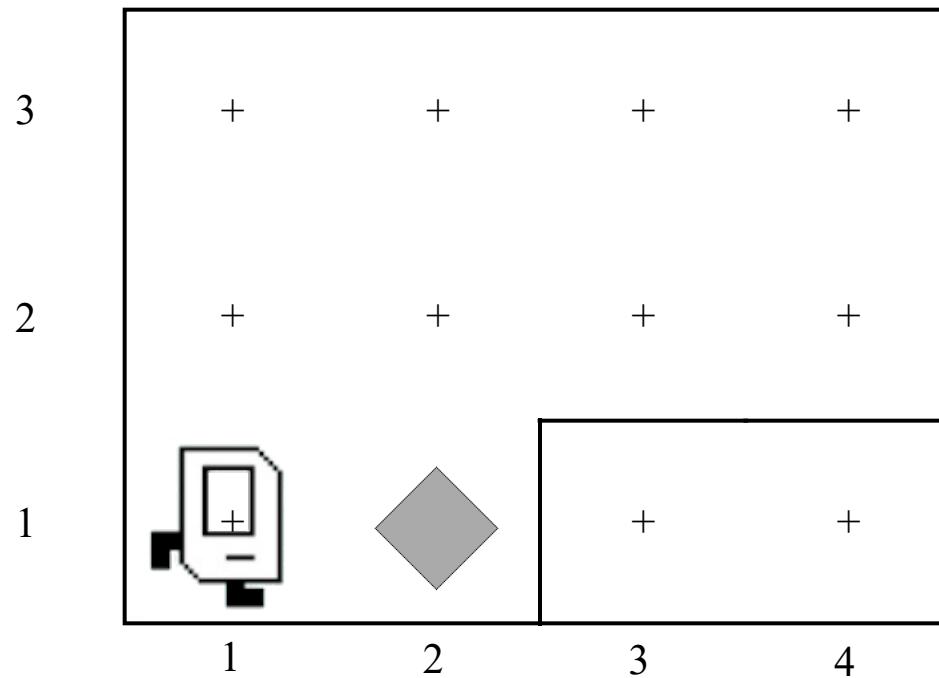


move();

bougez

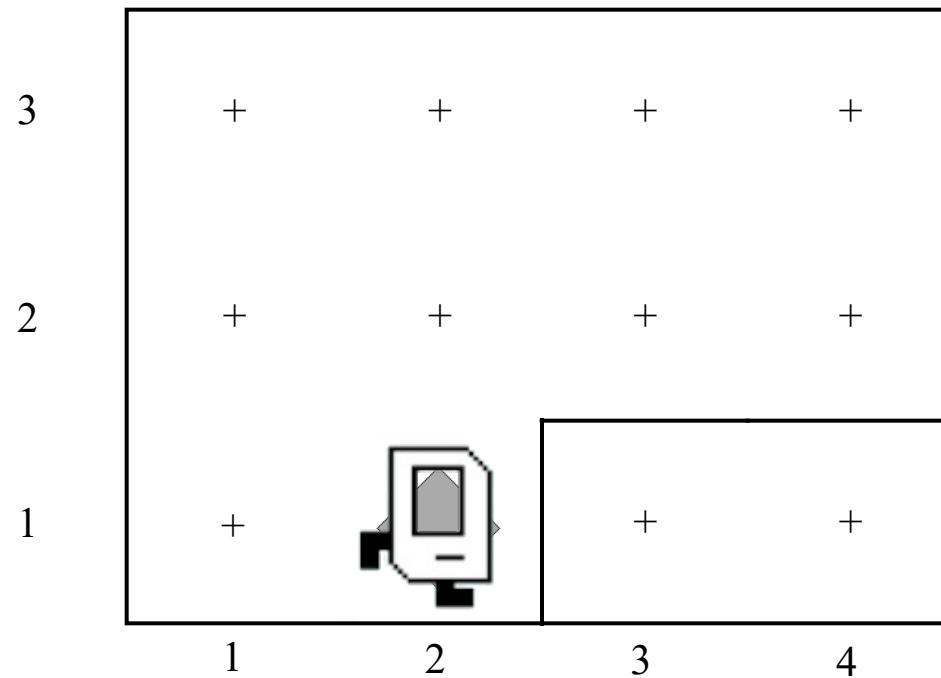
move();

bougez



move();

bougez

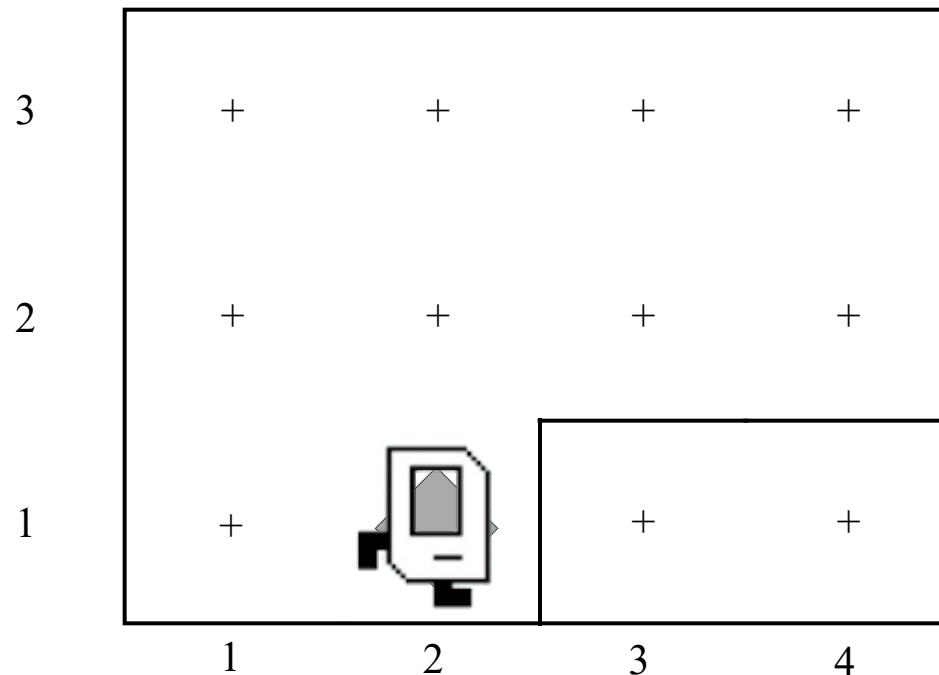


turnLeft();

tournez à gauche

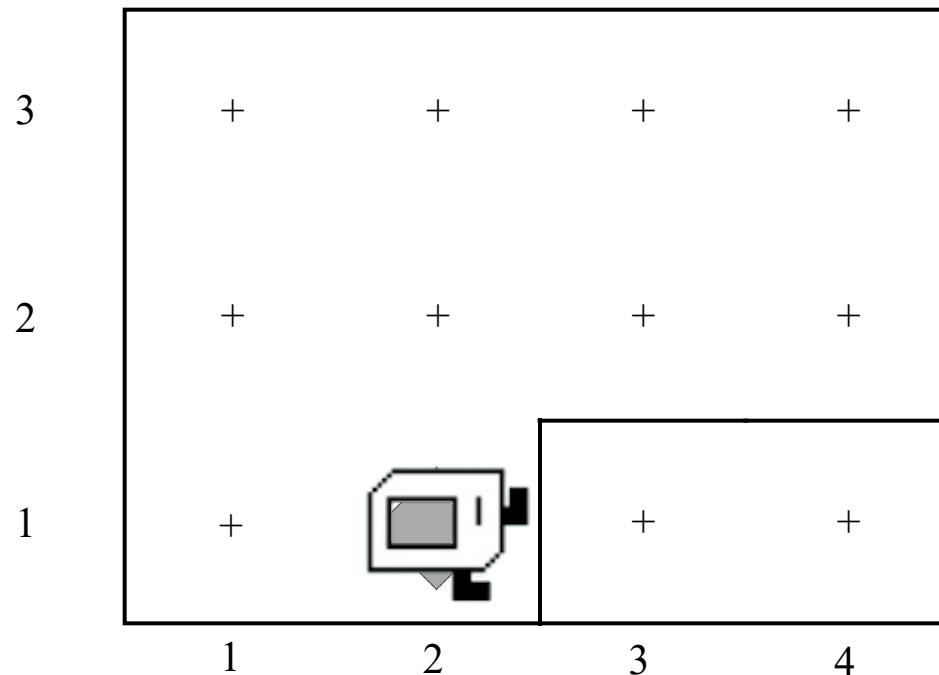
turnLeft();

tournez à gauche



turnLeft();

tournez à gauche

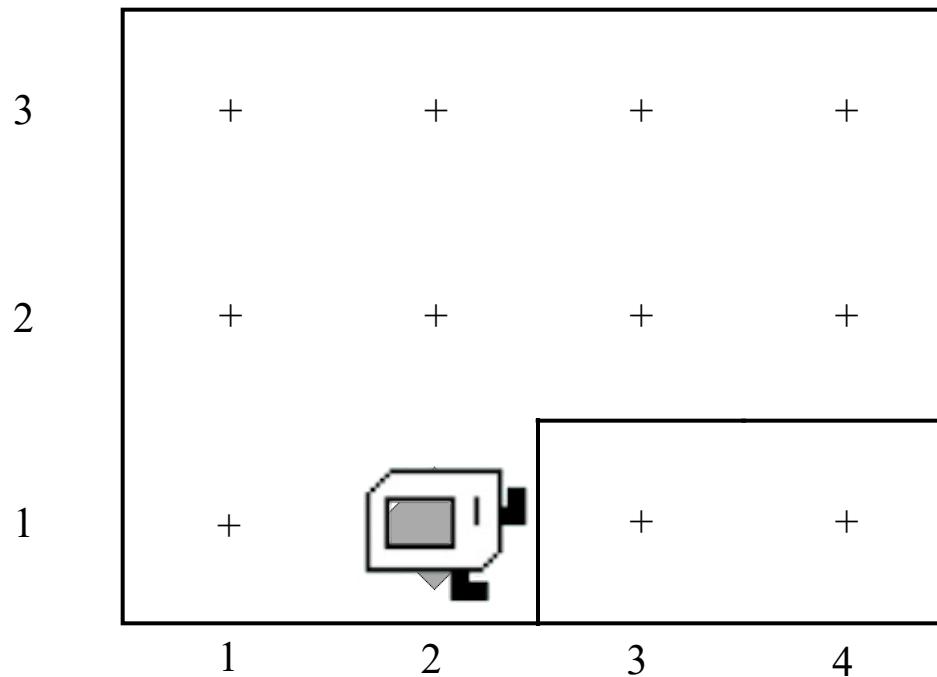


pickBeeper();

prendre le bip

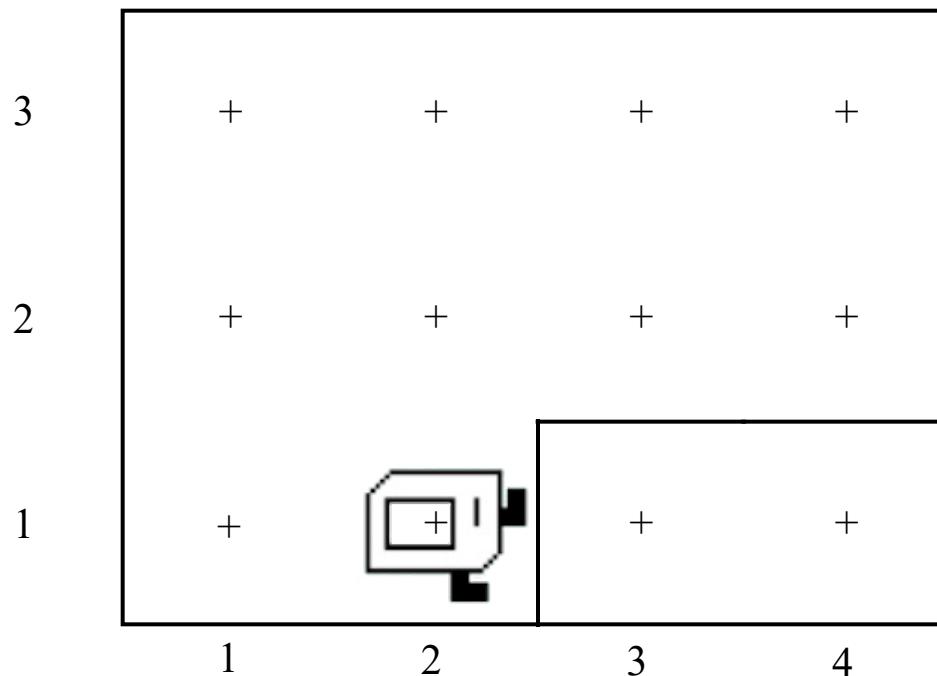
pickBeeper();

prendre le bip



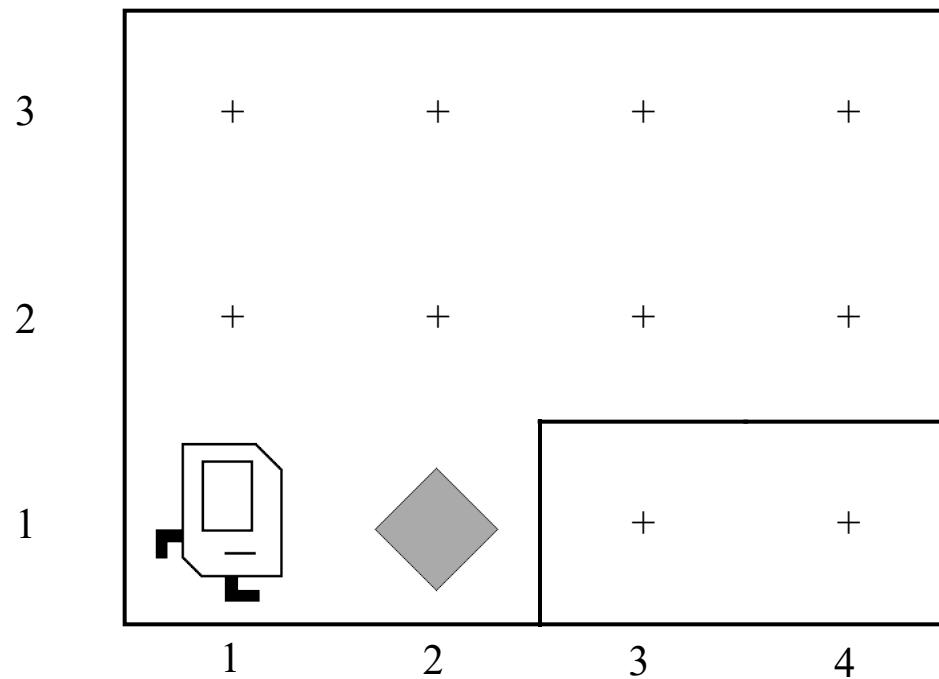
pickBeeper();

prendre le bip

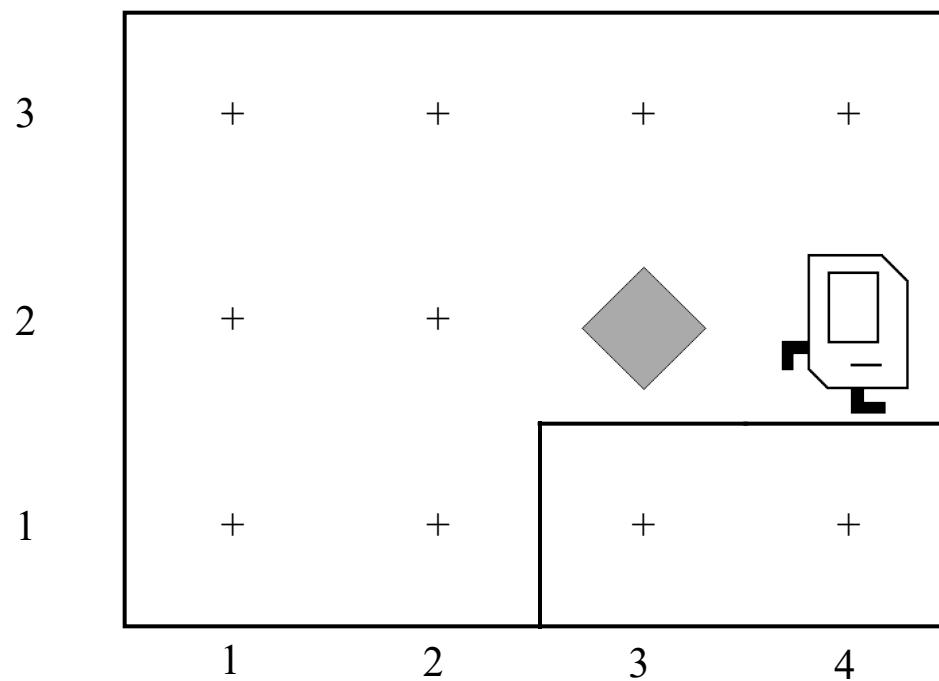


Questions?

Premier Défi



Premier Défi



J'ai besoin d'un volontaire



Essayons

eclipse

Anatomie d'un Programme

```
import stanford.karel.*;  
  
public class OurKarelProgram extends Karel {  
    public void run() {  
        \\ Si j'écris ici, cela ne fait pas  
        partie du code. Ce sont des notes pour  
        vous-même et pour les autres qui  
        souhaitent lire votre programme.  
    }  
  
    private void tournezADroite() {  
        turnLeft();  
        turnLeft();  
        turnLeft();  
    }  
}
```

Anatomie d'un Programme

```
import stanford.karel.*;  
  
public class OurKarelProgram extends Karel {  
    public void run() {  
        \\ Si j'écris ici, cela ne fait pas  
        partie du code. Ce sont des notes pour  
        vous-même et pour les autres qui  
        souhaitent lire votre programme.  
    }  
  
    private void tournezADroite() {  
        turnLeft();  
        turnLeft();  
        turnLeft();  
    }  
}
```

COMMENTAIRE

```
import stanford.karel.*;

public class OurKarelProgram extends Karel {
    public void run() {
        \\ Si j'écris ici, cela ne fait pas
        partie du code. Ce sont des notes pour
        vous-même et pour les autres qui
        souhaitent lire votre programme.
        On dit c'est le COMMENTAIRE.

    }
}
```

Le Style

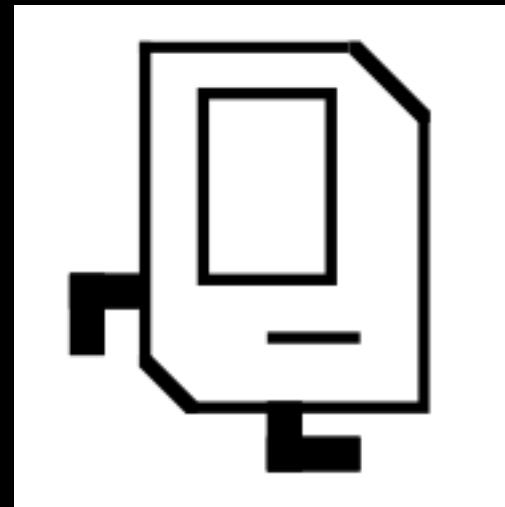
```
public void run() {  
    move();  
    pickBeeper();  
    turnLeft();  
    move();  
    turnLeft();  
    turnLeft();  
    turnLeft();  
    move();  
    putBeeper();  
    move();  
}
```

```
public void run() {  
    move();  
    turnLeft();  
    move();  
    turnLeft();  
    turnLeft();  
    turnLeft();  
    turnLeft();  
    turnLeft();  
    turnLeft();  
    move();  
    move();  
    pickBeeper();  
    turnLeft();  
    turnLeft();  
    move();  
    move();  
    turnLeft();  
    turnLeft();  
    turnLeft();  
    turnLeft();  
    move();  
    turnLeft();  
    turnLeft();  
    turnLeft();  
    turnLeft();  
    turnLeft();  
    move();  
    putBeeper();  
    turnLeft();  
    move();  
}
```

Questions?

Nous ne voulons pas toujours dit à Karel de tourner à gauche trois fois.

On veux juste l'apprendre à tourner à droite..



Nous pouvons enseigner de nouvelles choses
à Karel.

Vous avez vu “public void”.

```
import stanford.karel.*;
/*
 * Programme: Step Up
 *
 * Votre premier exemple de programme Karel. Demandez à Karel de prendre le bip devant vous
 * d'elle et placez-le sur le rebord.
 * Ceci est un commentaire. Votre ordinateur l'ignorera.
 */
public class StepUp extends Karel {

    // Lorsque vous démarrez votre programme, ce code sera exécuté.
    public void run() {
        move();
        pickBeeper();
        turnLeft();
        move();
        turnLeft();
        turnLeft();
        turnLeft();
        move();
        putBeeper();
        move();
    }
}
```

Nous pouvons enseigner de nouvelles méthodes à
Karel en écrivant

“private void”

Vous avez vu “public void”.
Nous pouvons enseigner de nouvelles méthodes à
Karel en écrivant

“private void”

Vous avez vu “public void”.
Nous pouvons enseigner de nouvelles
METHODES à Karel en écrivant

“private void”

Vous avez vu “public void”.
Nous pouvons enseigner de nouvelles
METHODES à Karel en écrivant

“private void”

On appelle ça DECOMPOSITION

Définition de la méthode

```
private void nom( ) {  
    declarations dans le corps de  
    la méthode  
}
```

Cela ajoute une
nouvelle commande
au vocabulaire de
Karel

On va voir avec ce qu'on a déjà fait avec Karel..
Qui a une idée comment on peut le décomposer?

Décomposition

```
import stanford.karel.*;

public class
OurKarelProgram extends
Karel {
    public void run() {
        move();
        pickBeeper();
        move();
        turnLeft();
        move();
        turnLeft();
        turnLeft();
        turnLeft();
        move();
        putBeeper();
        move();
    }
}
```

```
import stanford.karel.*;

public class OurKarelProgram
extends Karel {
    public void run() {
        move();
        pickBeeper();
        move();
        turnLeft();
        move();
        turnezADroite();
        move();
        putBeeper();
        move();
    }

    private void turnezADroite(){
        turnLeft();
        turnLeft();
        turnLeft();
    }
}
```

Maintenant, nous avons appris à Karel une nouvelle compétence.

Maintenant, elle sait comment tourner à droite.

Chaque fois que nous lui dirons de tourner à droite, elle se référera à cette méthode.

Décomposition

```
import stanford.karel.*;  
  
public class  
OurKarelProgram extends  
Karel {  
    public void run() {  
        move();  
        pickBeeper();  
        move();  
        turnLeft();  
        move();  
        turnLeft();  
        turnLeft();  
        turnLeft();  
        move();  
        putBeeper();  
        move();  
    }  
}
```

```
import stanford.karel.*;  
  
public class OurKarelProgram  
extends Karel {  
    public void run() {  
        move();  
        pickBeeper();  
        move();  
        turnLeft();  
        move();  
        turnnezADroite();  
        move();  
        putBeeper();  
        move();  
    }  
  
    private void turnnezADroite(){  
        turnLeft();  
        turnLeft();  
        turnLeft();  
    }  
}
```

Anatomie d'un Programme

```
import stanford.karel.*;

public class OurKarelProgram extends Karel {
    public void run() {
        move();
        pickBeeper();
        move();
        turnLeft();
        move();
        turnezADroite();
        move();
        putBeeper();
        move();
    }

    private void turnezADroite() {
        turnLeft();
        turnLeft();
        turnLeft();
    }
}
```

Anatomie d'un Programme

```
import stanford.karel.*;  
  
public class OurKarelProgram extends Karel {  
    public void run() {  
        move();  
        pickBeeper();  
        move();  
        turnLeft();  
        move();  
        turnezADroite();  
        move();  
        putBeeper();  
        move();  
    }  
  
    private void turnezADroite() {  
        turnLeft();  
        turnLeft();  
        turnLeft();  
    }  
}
```

Ceci est le **code source**
du programme

Anatomie d'un Programme

```
import stanford.karel.*;  
  
public class OurKarelProgram extends Karel {  
    public void run() {  
        move();  
        pickBeeper();  
        move();  
        turnLeft();  
        move();  
        turnezADroite();  
        move();  
        putBeeper();  
        move();  
    }  
}
```

Cette partie du *code source*
s'appelle une *méthode*.

```
private void turnezADroite(){  
    turnLeft();  
    turnLeft();  
    turnLeft();  
}  
}
```

Anatomie d'un Programme

```
import stanford.karel.*;  
  
public class OurKarelProgram extends Karel {  
    public void run() {  
        move();  
        pickBeeper();  
        move();  
        turnLeft();  
        move();  
        turnezADroite();  
        move();  
        putBeeper();  
        move();  
    }  
  
    private void turnezADroite(){  
        turnLeft();  
        turnLeft();  
        turnLeft();  
    }  
}
```

Cette ligne de code donne le
nom de la méthode
(ici, run)

Anatomie d'un Programme

```
import stanford.karel.*;  
  
public class OurKarelProgram extends Karel {  
    public void run() {  
        move();  
        pickBeeper();  
        move();  
        turnLeft();  
        move();  
        turnezADroite();  
        move();  
        putBeeper();  
        move();  
    }  
  
    private void turnezADroite(){  
        turnLeft();  
        turnLeft();  
        turnLeft();  
    }  
}
```

Cette ligne de code donne le
nom de la méthode
(ici, `turnezADroite`)

Anatomie d'un Programme

```
import stanford.karel.*;  
  
public class OurKarelProgram extends Karel {  
    public void run() {  
        move();  
        pickBeeper();  
        move();  
        turnLeft();  
        move();  
        turnezADroite();  
        putBeeper();  
        move();  
    }  
}
```

Ceci s'appelle une ***declaration d'importation***. Il dit à Java ce que Karel est.

```
private void turnezADroite() {  
    turnLeft();  
    turnLeft();  
    turnLeft();  
}  
}
```

Anatomie d'un Programme

```
import stanford.karel.*;  
  
public class OurKarelProgram extends Karel {  
    public void run() {  
        move();  
        pickBeeper();  
        move();  
        turnLeft();  
        move();  
        turnezADroite();  
        move();  
        putBeeper();  
        move();  
    }  
  
    private void turnezADroite(){  
        turnLeft();  
        turnLeft();  
        turnLeft();  
    }  
}
```

Ceci s'appelle un
bloc de code.
**Signifier par des
intervals {}.**

Style de Programme

```
import stanford.karel.*;
```

```
public class OurKarelProgram extends Karel {  
    public void run() {  
        move();
```

```
        pickBeeper();  
        move();
```

```
        turnLeft();
```

```
        move();  
        turnRight();  
        move();
```

```
        putBeeper();
```

```
        move();
```

```
}
```

```
    private void turnezADroite() {  
        turnLeft();
```

```
        turnLeft();
```

```
        turnLeft();
```

```
}
```

```
}
```

Style de Programme

```
import stanford.karel.*;

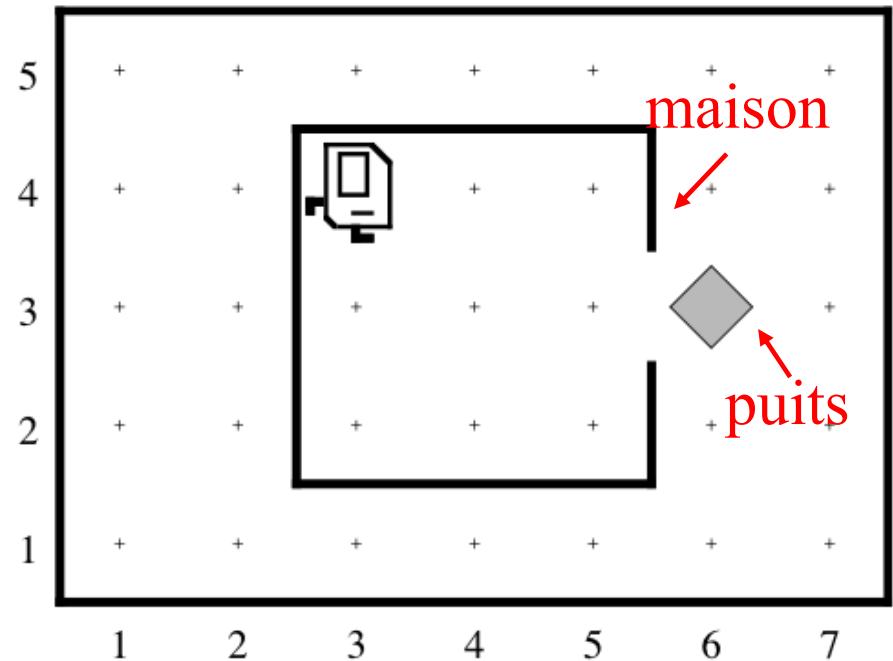
public class OurKarelProgram extends Karel {
    public void run() {
        move();
        pickBeeper();
        move();
        turnLeft();
        move();
        turnezADroite();
        move();
        putBeeper();
        move();
    }

    private void turnezADroite(){
        turnLeft();
        turnLeft();
        turnLeft();
    }
}
```

EXERCISE: Commander Karel de puisez de l'eau pour toi



EXERCISE: Puissez de l'eau



Décomposer dans trois méthodes:

1. Passer au puits,
2. Puissez l'eau et
3. Retour à son point de départ.

Elle connaît quatre commandes:

move(); bougez

turnLeft(); tournez à gauche

putBeeper(); mettre le bip

pickBeeper(); prendre le bip

