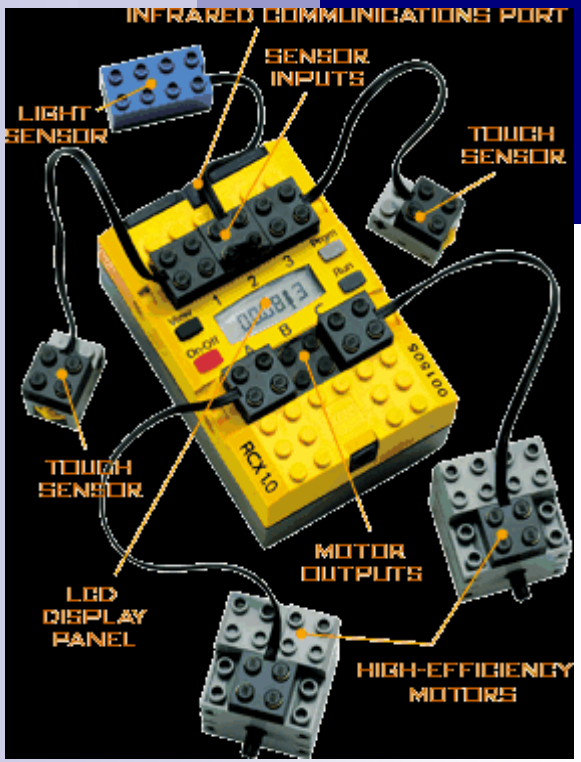
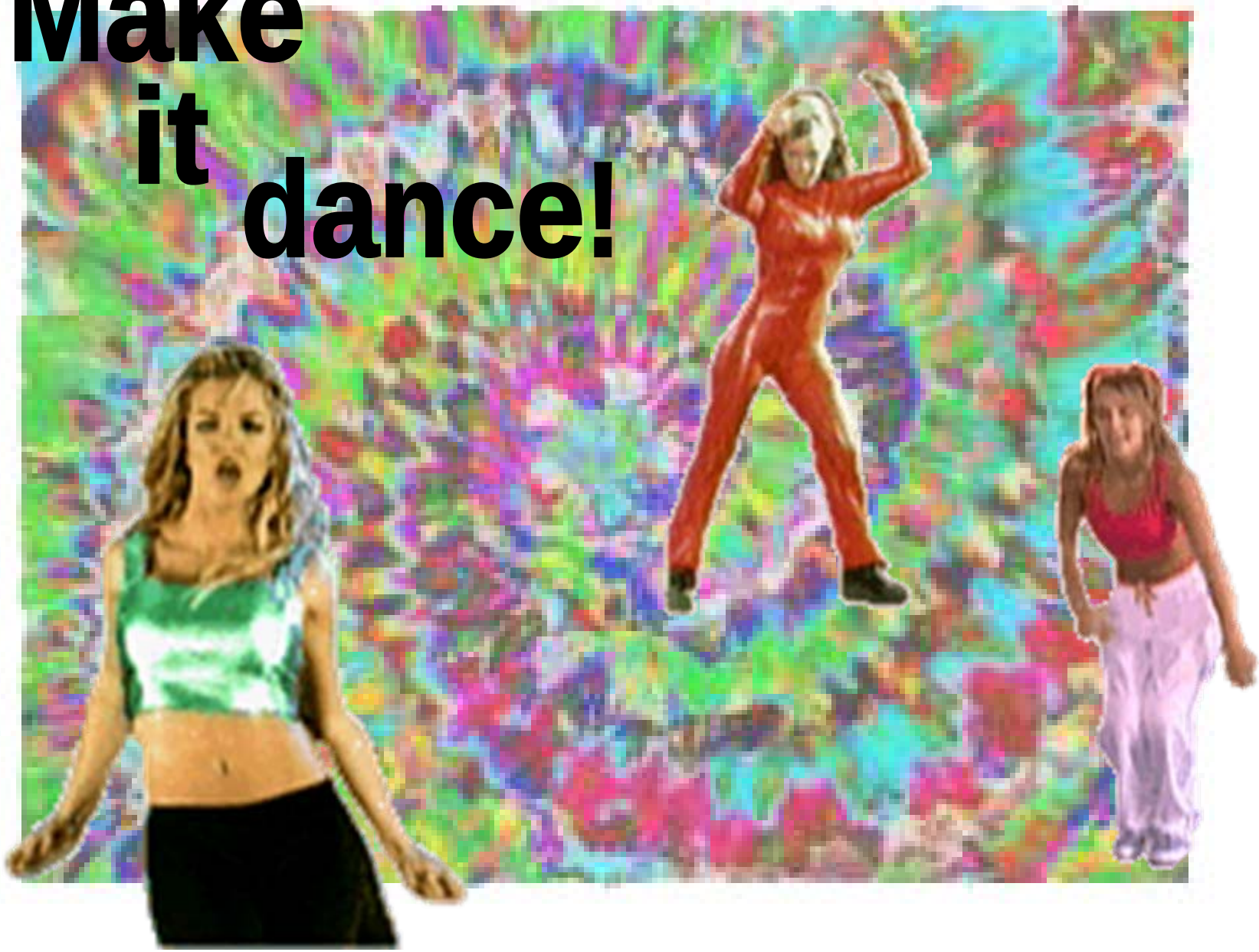
Making Robots Move



So you have a robot.

Now what?

Make it dance!



Command conversion

The commands you can use to program your robot are very similar to Javascript commands

I want my robot to…	LEGO commands
Go forward!	forward(10);
Go backwards!	back(10);
Rotate right!	right(90);
Rotate left!	left(45);

Repeat commands	for(i=0; i<4; i++)
	{
	command;
	}

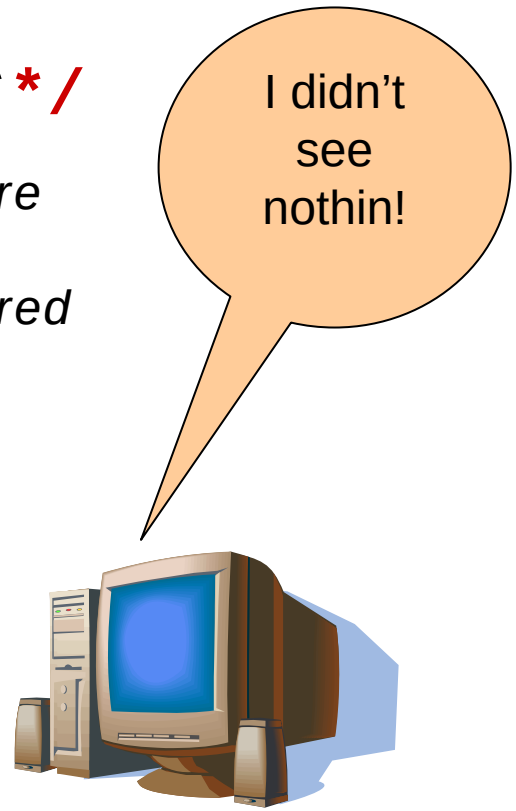
For example: if you want your robot to make a square, you first tell it to go forward and then tell it to rotate 90 degrees. Repeat 4 times, and you have a square!

When you program, you should…

Use COMMENTS!

```
/*************************************
*          SLASHES AND STARS          *
*************************************/
```

The computer will ignore anything between the slashes and stars (the red ones)



You use comments to make notes to yourself in English when you're programming.

You can see our comments in the files we've given to you.

AHHHHHHH!

Let’s break that down a little bit…

Things to ignore …

```
#include "wrapper.h"


int main()
{
   …

      return 0;
}
```

This connects the file you make with what we've already written

The main line tells your computer where to start. All your code goes in between the two curly braces under main().

Returning a zero at the last line tells the computer the program is over.

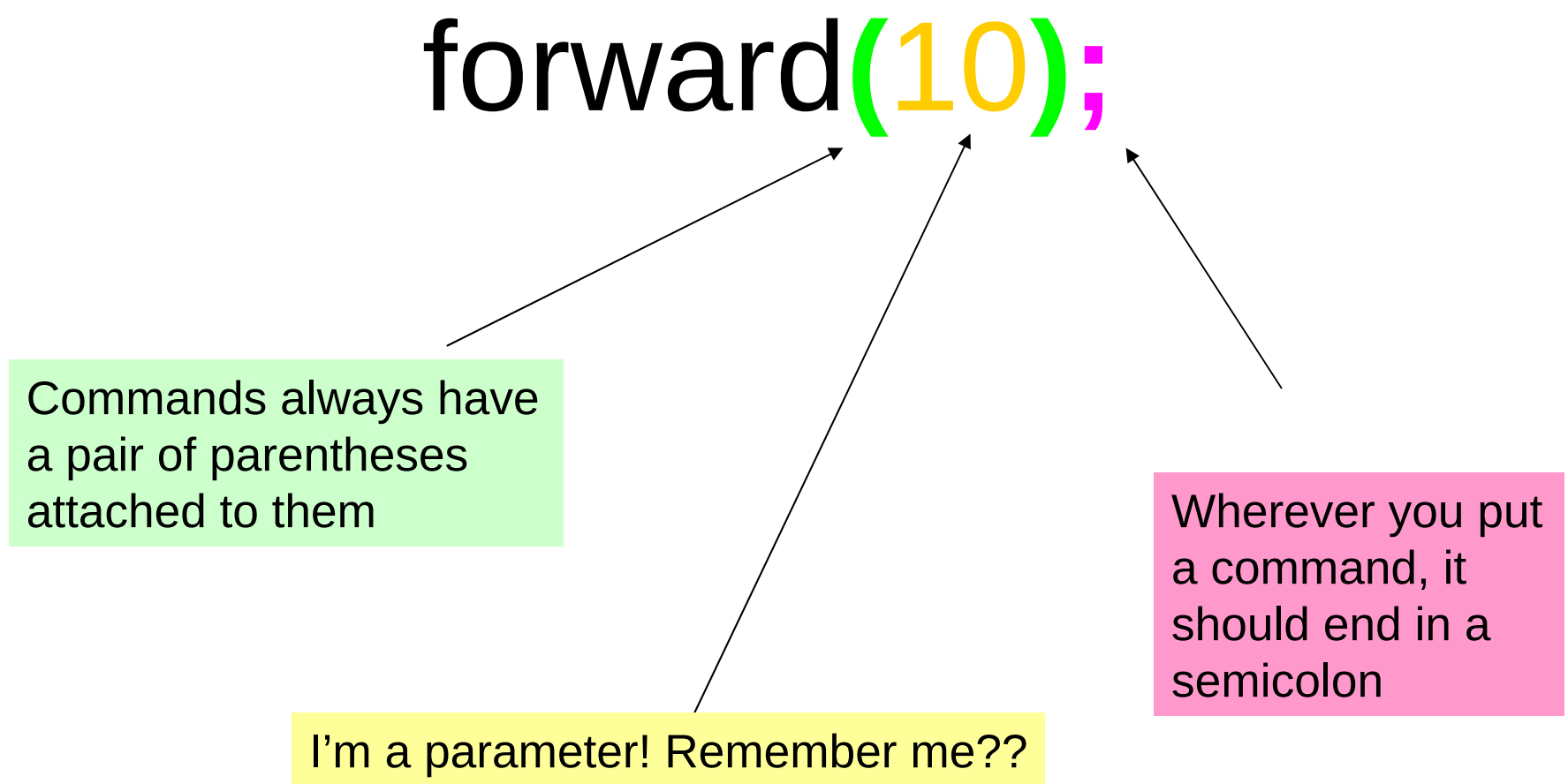
A really simple program

```
#include "./support/logo.h"
int main()
{
        setSpeed(10);
        forward(3);
        return 0;
}
```

←Set the motors at the highest speed

←Go forward for 3 seconds

Important stuff:



Important stuff continued:

!!!!!!REALLY IMPORTANT STUFF!!!!!!

Before you give the robot any movement commands, such as forward(10), back(10), etc, you need to set the speed of the motors using the setSpeed(#) command where # is a number from 1-10. If you don't do this, the speed will automatically be set to 0 and your movement commands will not work.

MORE important stuff:

```
if( touched(1) )
{
    forward(10);
}
```

I'm a conditional! Remember me??

`touched(1)` is a method which tells you if touch sensor 1 has been pressed.
**For more special methods, see your handout*

Curly braces are necessary when you are writing conditional statements

Let's look at something harder:

```
#include "wrapper.h"

int i;

int main(int argc, char *argv[ ])
{
        setSpeed(10);

        while(1)
        {
                forward(10);
                wait(2);

                if ( touched(1) )
                {
                        setSpeed(10);
                        rightCircle(10);
                        wait(2);
                        setSpeed(6);
                        back(10);
                        wait(3)
                }
                else
                {

                        setSpeed(10);
                        leftCircle(10);
                        wait(10);
                }
        }
        return 0;
}
```

You've created a program and it works (yay!)
…but your robot doesn't move *quite* as it should.

Now what?

kay, so it's not as easy as pie…

Like humans food for energy, robots need batteries.

If you eat a big breakfast, and we say, "Run for 12 minutes", you might run a mile.

If you didn't eat, and we say, "Run for 12 minutes", you might only go half a mile.

Robots are the same. Telling it to turn 360^{o} is actually telling it to turn for an amount of time. If the battery is dying, it might take 400^{o} for a circle.

Now we're going to…

- Write a few simple programs to get an ideas of how things work.
- Choreograph a dance routine
- Program the dance into your robot
- Let your robot show its stuff!