# Last Time

- Basics of programming
  - *create source code, compile, execute* (repeat)
- Basics of programming in Java
  - *case-sensitive, standard framework, statements ending with ";", groups enclosed in "{…}"*
- Basics of using Eclipse to program in Java
  - *projects and classes, automatic spell-check, hint(s) to fix mistakes, running a program*
- How to use **named components**
  - *LCD, Sound, Motor, Button*

# The “LCD” Component

This component supports the functions:

- `LCD.clear();`
- `LCD.drawChar(char c, int x, int y);`
- `LCD.drawInt(int i, int x, int y);`
- `LCD.drawString(String s, int x, int y);`
- `LCD.refresh();`

# The “Sound” Component

This component supports the functions:

- `Sound.beep();`
- `Sound.beepSequence();`
- `Sound.beepSequenceUp();`
- `Sound.buzz();`
- `Sound.pause(millisecs);`
- `Sound.playTone(freq, duration);`

# The “Button” Component

This component contains the subcomponents

ESCAPE, ENTER, LEFT, RIGHT

which in turn support the functions:

- `isUp();`
- `isDown();`
- `waitForPress();`
- `waitForPressAndRelease();`

## The “Motor” Component

This component contains subcomponents A, B, and C, which in turn support the functions:

- `backward();`
- `forward();`
- `flt();`
- `isMoving();`
- `getTachoCount();`
- `resetTachoCount();`
- `rotate(int angle)`
- `rotate(int angle, boolean returnImmediately)`
- `setAcceleration(int acc)`
- `setSpeed(int speed)`

SETON HALL UNIVERSITY

# Example: Play Music (1)

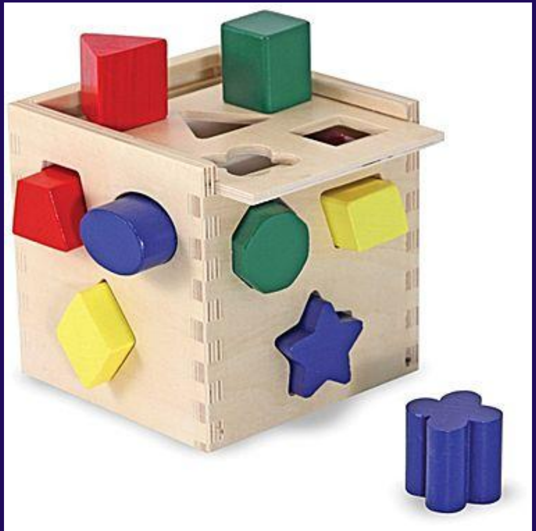
```
public class PlayBeethoven
{
   public static void main(String args[])
   {
      // play “e” three times
      Sound.playTone(659, 200);
      Sound.pause(220);

      Sound.playTone(659, 200);
      Sound.pause(220);

      Sound.playTone(659, 200);
      Sound.pause(220);

      // play “c”
      Sound.playTone(523, 600);
      Sound.pause(600);
   }
}
```

## Variables

You can define variables to hold data of a specific type:

- `int` (an integer)
- `float` (a "small" decimal)
- `double` (a decimal)
- `boolean` (true or false)
- `String` (of characters)

*(a variable is a "bucket" that can hold some specific kind of data)*



```
int number = 10;
```

## Example: Play Music (2)

```
public class PlayBeethoven
{
   public static void main(String args[])
   {
      int E = 659;
      int C = 523;

      Sound.playTone(E, 200);
      Sound.pause(220);
      Sound.playTone(E, 200);
      Sound.pause(220);
      Sound.playTone(E, 200);
      Sound.pause(220);
      Sound.playTone(C, 600);
      Sound.pause(600);
   }
}
```

# Special variables: Constants

Sometimes variables don’t vary but provide a convenient name for a value that won’t change

- *define constants of a specific type immediately after the “class” and before the “main” method*
- *can do calculations with constants and variables*

**Example**:

```
static final double PI = 3.1415;
static final NAME = “Bert”;

      ...
double r = 3.0;
double circleArea = PI * r*r;
```

# Example: Play Music (3)

```
public class PlayBeethoven
{
   static final int E = 659;
   static final int C = 523;
   static final int TIME = 200;

   public static void main(String args[])
   {
      Sound.playTone(E, TIME);
      Sound.pause(TIME);
      Sound.playTone(E, TIME);
      Sound.pause(TIME);
      Sound.playTone(E, TIME);
      Sound.pause(TIME);
      Sound.playTone(C, 600);
      Sound.pause(600);
   }
}
```

# Variables and Computations

Java provides the following operators for computations:

- `+` (addition)
- `-` (subtraction)
- `*` (multiplication)
- `/` (division)
- `%` (remainder after integer division)

Results of computations can be assigned to a variable or used as input to functions

`double r = (10 % 3)` (= is assignment op)

# Example: Play Music (4)

```
public class PlayBeethoven
{
   static final int E = 659;
   static final int C = 523;
   static final int TIME = 200;

   public static void main(String args[])
   {
      Sound.playTone(E, TIME);
      Sound.pause(TIME + 50);
      Sound.playTone(E, TIME);
      Sound.pause(TIME + 50);
      Sound.playTone(E, TIME);
      Sound.pause(TIME + 50);
      Sound.playTone(C, 3*TIME);
      Sound.pause(3*(TIME + 50));
   }
}
```

# Functions

- Frequently some lines of code can be combined into functional units called “functions” (or “methods”)
- Every function has a *name*, a *return type*, and an *(optional) input list*, collectively called the *function header*, as well as a *function body*. Once defined, functions  can be used multiply times
- Functions are defined before the “`main`” function

- ***Clever and flexible definitions of functions are the hallmark of any good program!!!***

## Example: Play Music (5)

```
public class PlayBeethoven
{
   static final int E = 659;
   static final int C = 523;
   static final int TIME = 200;

   public static void play(int freq, int duration)
   {
      Sound.playTone(freq, duration);
      Sound.pause(duration + 50);
   }

   public static void main(String args[])
   {
      play(E, TIME);
      play(E, TIME);
      play(E, TIME);
      play(C, 3*TIME);
   }
}
```

# Mandatory Comments

- Every program must contain comments for the following:
  - the programmer's **name** (use @author)
  - the **date** or **version** when the program was created (use @version)
  - a brief **description** in English as to what the program does

  - Any defined function should include a comment explaining what it does and what the input and output of the function is

# Example: Play Music (6)

```
/*
 * This program plays the first few notes of Beethoven's 5th symphony
 *
 * @author Bert Wachsmuth
 * @version 1.0 (01/27/2014)
 */
public class PlayBeethoven
{
   // defining the frequencies of the notes used
   static final int E = 659;
   static final int C = 523;
   // defining the base length of a note
   static final int TIME = 200;

   // function to play a note at a given frequency and duration
   public static void play(int freq, int duration)
   {
      ... Rest as before ...
}
```

# Robot Task 1

- Create a program to play a “song”, where “song” is defined as a collection of **at least 4 notes**
- Your program must include **variables** or **constants** or both as well as **functions**
- Your program must include **comments** for your **name**, the **version** or **date**, and a **brief** program **description**
- EXTRA: Your program can show the *name* and *composer* of the song on the LCD panel while playing the song

- You need to *submit the printed program as well as demonstrate it* (i.e. play the song)