動画の作成1

担当: 佐藤

計算機実習III

第3回

sato@ise.aoyama.ac.jp

2019/4/23

setup()とdraw()の振る舞い

sato@ise.aoyama.ac.jp

2019/4/23

draw()

- Processingでは，draw()の中のコードが繰り返し実行される
  - 通常，draw()はsetup()と共に用いる
    - size()をdraw()の中に書くとエラー．setup()の冒頭に書く
  - draw()の中のコードの実行が一巡する単位をフレームという
    - 60[frame/sec](デフォルト)

例

フレームの実行回数（システム変数）

```
void draw() {
  System.out.printf("%4d", frameCount);
  if (frameCount % 10 == 0) {
    println("");
    if (frameCount % 60 == 0) {
      println("the elapsed time[sec]: " + millis() / 1000);
    }
  }
  if (frameCount / 60 >= 3) {
    noLoop();
  }
}
```

スケッチの実行開始時からの経過時間[msec]

draw()の繰り返しを止める

```
   1   2   3   4   5   6   7   8   9  10
  11  12  13  14  15  16  17  18  19  20
  21  22  23  24  25  26  27  28  29  30
  31  32  33  34  35  36  37  38  39  40
  41  42  43  44  45  46  47  48  49  50
  51  52  53  54  55  56  57  58  59  60
the elapsed time[sec]: 1
  61  62  63  64  65  66  67  68  69  70
  71  72  73  74  75  76  77  78  79  80
  81  82  83  84  85  86  87  88  89  90
  91  92  93  94  95  96  97  98  99 100
 101 102 103 104 105 106 107 108 109 110
 111 112 113 114 115 116 117 118 119 120
the elapsed time[sec]: 2
 121 122 123 124 125 126 127 128 129 130
 131 132 133 134 135 136 137 138 139 140
 141 142 143 144 145 146 147 148 149 150
 151 152 153 154 155 156 157 158 159 160
 161 162 163 164 165 166 167 168 169 170
 171 172 173 174 175 176 177 178 179 180
the elapsed time[sec]: 3
```

frameRate()

- **frameRate()**: 1秒間あたりのフレーム数を変更する関数
  - 引数: 1秒間あたりのフレーム数(**フレームレート**)
    - frameRate()の設定は厳密に反映されるとは限らない(CPU速度に依存)
  - 通常，setup()の中で用いる

例

```
void setup() {
  frameRate(30);
}

void draw() {
  System.out.printf("%4d", frameCount);
  if (frameCount % 10 == 0) {
    println("");
    if (frameCount % 60 == 0) {
      println("the elapsed time[sec]: " + millis() / 1000);
    }
  }
  if (frameCount / 60 >= 3) {
    noLoop();
  }
}
```

30[frame/sec]

```
   1   2   3   4   5   6   7   8   9  10
  11  12  13  14  15  16  17  18  19  20
  21  22  23  24  25  26  27  28  29  30
  31  32  33  34  35  36  37  38  39  40
  41  42  43  44  45  46  47  48  49  50
  51  52  53  54  55  56  57  58  59  60
the elapsed time[sec]: 2
  61  62  63  64  65  66  67  68  69  70
  71  72  73  74  75  76  77  78  79  80
  81  82  83  84  85  86  87  88  89  90
  91  92  93  94  95  96  97  98  99 100
 101 102 103 104 105 106 107 108 109 110
 111 112 113 114 115 116 117 118 119 120
the elapsed time[sec]: 4
 121 122 123 124 125 126 127 128 129 130
 131 132 133 134 135 136 137 138 139 140
 141 142 143 144 145 146 147 148 149 150
 151 152 153 154 155 156 157 158 159 160
 161 162 163 164 165 166 167 168 169 170
 171 172 173 174 175 176 177 178 179 180
the elapsed time[sec]: 6
```

モード

- Processingのスケッチは，スタティックモードかアクティブモードのいずれか一方で実行される
  - スタティックモード: 動きのないスケッチ(静止画)作成用
    - setup()とdraw()両方なし
    - setup()のみ
  - アクティブモード: 動きのあるスケッチ(動画)作成用
    - setup()とdraw()両方あり
    - draw()のみ
- アクティブモードの注意
  - 入出力関数(loadFont()など)は時間がかかる➔動作が重くなるためdraw()の中で使用しないよう注意！

インタラクティブな画像

sato@ise.aoyama.ac.jp

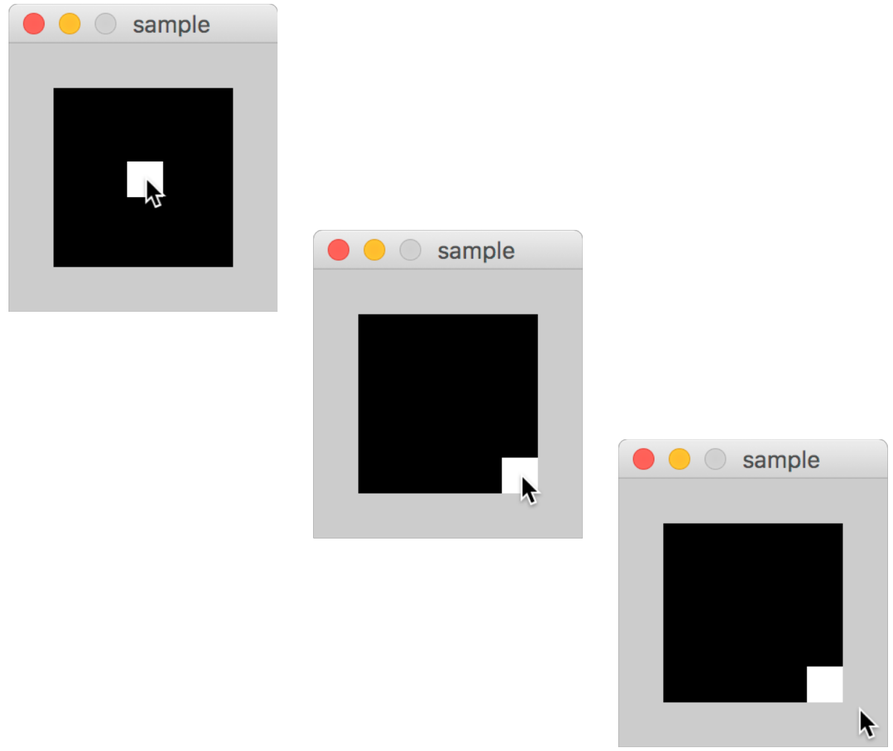
2019/4/23

mouseX，mouseY

- **mouseX**: ウィンドウ上のマウスポインタの**現在**位置のx座標を保持するint型のシステム変数
- **mouseY**: ウィンドウ上のマウスポインタの**現在**位置のy座標を保持するint型のシステム変数

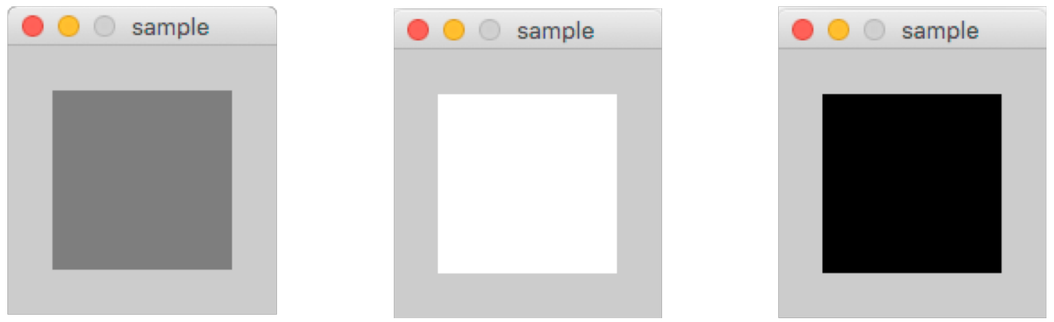
例

```
float d = 20; // diameter

void setup() {
  size(150, 150);
  rectMode(CENTER);
  noStroke();
}

void draw()
{
  background(204);
  fill(0);
  rect(width / 2, height / 2, 100, 100);
  float mX = constrain(mouseX, 25 + d / 2, 125 - d / 2);
  float mY = constrain(mouseY, 25 + d / 2, 125 - d / 2);
  fill(255);
  rect(mX, mY, d, d);
}
```



mousePressed，mouseButton

- mousePressed: マウスボタンが**押されている間**trueとなるboolean型のシステム変数
- mouseButton: **最後**に押されたボタンに対応する値を保持するint型のシステム変数
  - LEFT(左ボタン)
  - RIGHT(右ボタン)
  - CENTER(ホイール)



例

```
void setup() {
  size(150, 150);
  rectMode(CENTER);
  noStroke();
}
```

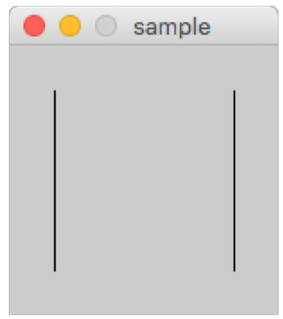
```
void draw() {
  if (mousePressed && mouseButton == LEFT) {
    fill(0);
  } else if (mousePressed && mouseButton == RIGHT) {
    fill(255);
  } else {
    fill(126);
  }
  rect(width / 2, height / 2, 100, 100);
}
```

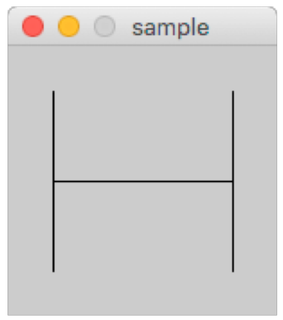
keyPressed，key

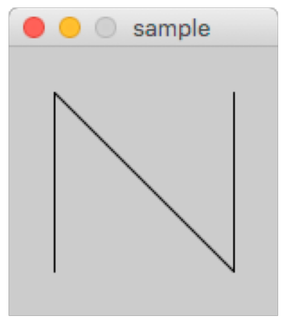
- keyPressed: キーボードのキーが押されている間trueとなるboolean型のシステム変数
- key: 最後に押されたASCIIコードの文字コードを保持するchar型のシステム変数
  - ASCIIコードに含まれる制御文字の文字コード
    - BACKSPACE，TAB，ENTER，RETURN，ESC，DELETE

例

```
void setup() {
  size(150, 150);
}
void draw() {
  background(204);
  line(25, 25, 25, 125);
  line(125, 25, 125, 125);
  if (keyPressed) {
    if (key == 'h' || key == BACKSPACE) {
      line(25, height / 2, 125, height / 2);
    }
    if (key == 'n' || key == DELETE) {
      line(25, 25, 125, 125);
    }
  }
}
```





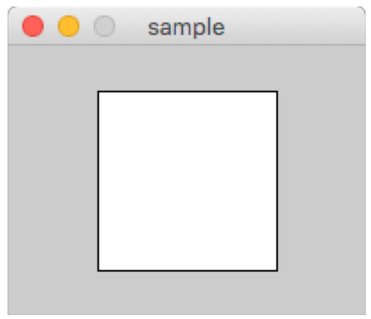


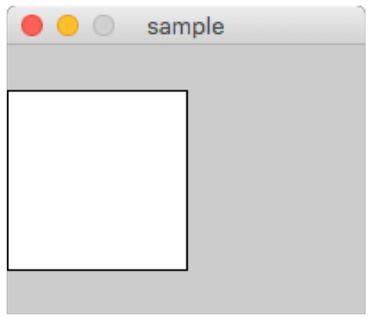
keyCode

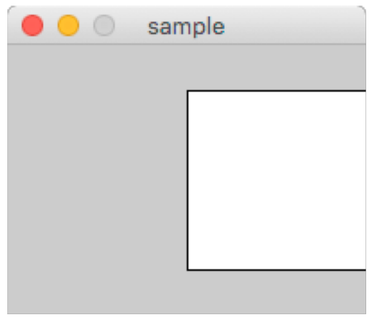
- keyCode: 最後に押された非ASCIIコードの特殊なキーに対応する値を保持するint型のシステム変数
  - 矢印キー: LEFT，RIGHT，UP，DOWN

例

```
float diam = 100;
float x;

void setup() {
  size(200, 150);
  x = width / 2;
  rectMode(CENTER);
}

void draw() {
  background(204);
  rect(x, height / 2, diam, diam);
  if (keyCode == LEFT) {
    x = max(diam / 2, --x);
  } else if (keyCode == RIGHT) {
    x = min(width - diam / 2, ++x);
  }
}
```







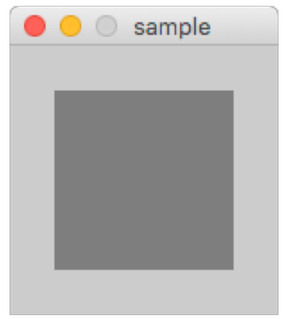
mousePressed()，mouseReleased()

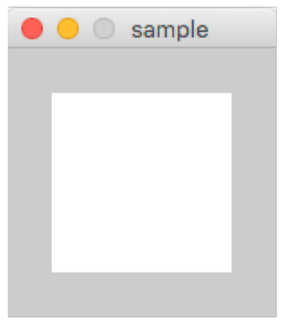
- mousePressed(): マウスボタンを押すと一度だけ呼ばれる関数
  - ボタンを押し続ける→mousePressed()が繰り返し呼ばれる
    - 繰り返しのタイミングはOS依存(フレームレート非依存)
  - アクティブモードでのみ動作する
- mouseReleased(): マウスボタンを離すと一度だけ呼ばれる関数
  - アクティブモードでのみ動作する

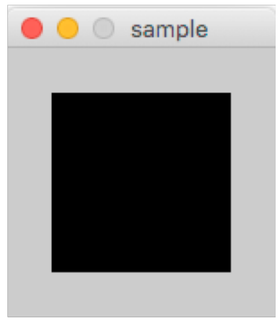
例

```
int value = 126;

void setup() {
  size(150, 150);
  rectMode(CENTER);
  noStroke();
}

void draw() {
  fill(value);
  rect(width / 2, height / 2, 100, 100);
}
```

```
void mousePressed() {
  if (mouseButton == LEFT) {
    value = 0;
  } else if (mouseButton == RIGHT) {
    value = 255;
  }
}

void mouseReleased() {
  value = 126;
}
```







keyPressed()

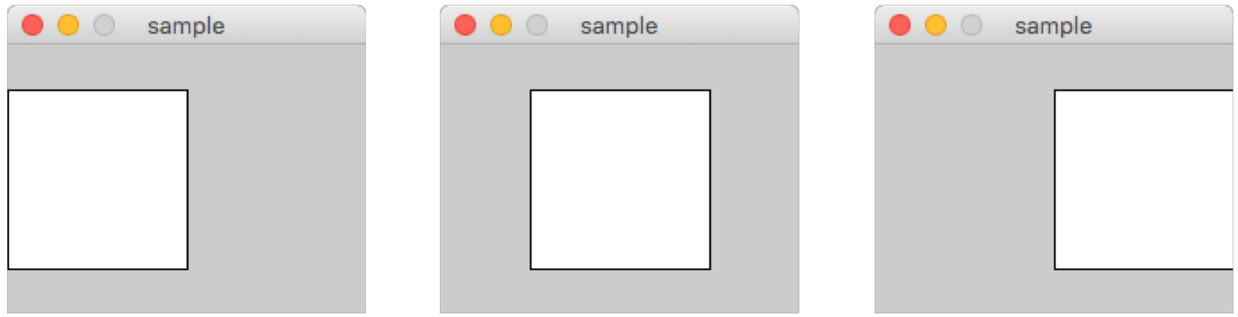
- **keyPressed(): キーボードのキーを押すと一度だけ呼ばれる関数**
  - **キーを押し続ける→keyPressed()が繰り返し呼ばれる．繰り返しのタイミングはOS依存(フレームレート非依存)**
  - **アクティブモードでのみ動作する**

例

```
float diam = 100;
float x;

void setup() {
  size(200, 150);
  x = width / 2;
  rectMode(CENTER);
}

void draw() {
  background(204);
  rect(x, height / 2, diam, diam);
}
```

```
void keyPressed() {
  if (keyCode == LEFT) {
    x = max(diam / 2, --x);
  } else if (keyCode == RIGHT) {
    x = min(width - diam / 2, ++x);
  }
}
```

キーが押された時のみxを更新



アクティブモードとイベント関数

- アクティブモードのスケッチがnoLoop()で停止していてもイベント関数は反応する

例

```
float elapsedTime = 0;
boolean canLoop = false;
int count = 0;
```

```
void mousePressed() {
  elapsedTime = millis();
  println("the mouse button was pressed.");
  count = 0;
  canLoop = true;
  loop();
}
```

draw()の繰り返しを始める

```
void setup() {
  noLoop();
}

void draw() {
  if (canLoop) {
    System.out.printf("%4d", ++count);
    if (count % 10 == 0) {
      println("");
      if (count % 60 == 0) {
        float exeTime = millis() - elapsedTime;
        println("the execution time[sec]: " + exeTime / 1000);
        // canLoop = false;
        noLoop();
      }
    }
  }
}
```

不要

```
the mouse button was pressed.
   1   2   3   4   5   6   7   8   9  10
  11  12  13  14  15  16  17  18  19  20
  21  22  23  24  25  26  27  28  29  30
  31  32  33  34  35  36  37  38  39  40
  41  42  43  44  45  46  47  48  49  50
  51  52  53  54  55  56  57  58  59  60
the execution time[sec]: 1.01
the mouse button was pressed.
   1   2   3   4   5   6   7   8   9  10
  11  12  13  14  15  16  17  18  19  20
  21  22  23  24  25  26  27  28  29  30
  31  32  33  34  35  36  37  38  39  40
  41  42  43  44  45  46  47  48  49  50
  51  52  53  54  55  56  57  58  59  60
the execution time[sec]: 0.989
```