LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Fakultät für Betriebswirtschaft
Munich School of Management

Basics in Programming for MMT

Session 2 – Control Structures

MMT

www.mmt.bwl.lmu.de

Scope of the Session

1. Theory
   - Colors
   - How to prevent crossing Borders?
   - If and Else
   - Boolean(-operators)
   - Slower than 1?
   - Float

2. Next
   - Session 3

3. Tutorial
   - Project

Theory

Theory

Colors

Black background	`background(0);`
White background	`background(255);`
RGB background	`background(r,g,b);`
Fill next objects red	`fill(255,0,0);` `rect(200,200,200,200);`
Set stroke color of next object to green	`stroke(0,255,0);` `rect(200,200,200,200);`

Theory

How to prevent crossing Borders? (1/2)

```
int x;
void setup () {
    size(600,600);
    x = 100;
}

void draw () {
    x = x+1;
    background(0);
    rect(x,200,200,200);
}
```

Theory

How to prevent crossing Borders? (2/2)

- **If** the rectangle moved further than a specific value along the x-axis, it should return.
- **Else** it can move further.

Theory

If and Else (1/4)

```
int x;
void setup () {
    size(600,600);
    x = 100;
}
void draw () {
    x = x+1;
```

If:
`x` crosses a certain point, the statements is true and the commands inside the `{...}` are executed.

```
    if (x>600);
        x = 0;
}
```

```
    background(0);
    rect(x,200,200,200);
}
```

Theory

If and Else (2/4)

Keyword

```
if
```

Condition:
A statement that describes a certain state
A statement is either `true` or `false`.

```
(x>0)
```

Consequence:
If the condition is `true`, the included commands are performed and otherwise skipped.

```
{...}
```

Theory

If and Else (3/4)

```
if (x>0) {...}
```

Keyword `else`

Consequence:
If the initial condition is `false`, the included commands are related to `else` are executed.

```
{...}
```

Theory

If and Else (4/4)

```
int x;
void setup () {
    size(600,600);
    x = 100;
}
void draw () {
    x = x+1;
```

Else:
The rectangle is just filled blue if `x` is smaller or equal 300.

```
    if (x>300);
         fill(255,0,0);
     }
     else {
         fill(0,0,255);
     }
```

```
background(0);
    rect(x,200,200,200);
}
```

Theory

Boolean(-operators) (1/4)

- Statements are either `true` or `false`.
- This binary data-type is called `boolean`.
- A **variable** can hold a boolean value or the **result of an operation** can be a boolean.

Theory

Boolean(-operators) (2/4)

`boolean`	`b`	`=`	`true`	`;`
Type	Name	Assignment	Value	End

Theory

Boolean(-operators) (3/4)

Statements generate boolean values.

greater than	x > 100
greater or equal	x >= 100
equal	x == 100
smaller or equal	x <= 100
smaller than	x < 100
unequal	x != 100

Theory

Boolean(-operators) (4/4)

Booleans can be **combined** or **manipulated** to new boolean values.

and	`true && true == true`
	`true && false == false`
	`false && true == false`
	`false && false == false`

or	`true || true == true`
	`true || false == true`
	`false || true == true`
	`false || false == false`

not	`!true == false` `!false == true`

Theory

Slower than 1?

Can we make the rectangle move slower than 1 pixel per draw step?

Theory

Float (1/2)

Integer:
Values such as 1 or 2 or …

```
int i = 100;
```

Float:
Values such as 0.2 or 1.22 or …

```
float f = 0.7;
```

Theory

Float (2/2)

```
float f;
void setup () {
    size(600,600);
    f = 0;
}

void draw () {
    background(0);
```

`f` is now increased in steps less than one.

```
    f = f + 0.3;
```

```
    rect(f,200,200,200);
}
```

Next

Next

Session 3

- Repeat instructions automatically
- Beyond numbers
- List of values

Tutorial

Tutorial

Project

1. How to make a circle **bounce off** the walls?
2. Think of using a **variable** to indicate the current **direction** of the circle
3. Implement bounce and movement in x and y direction