Čas br.2

Pojmovi:

- Blok koda
- Funkcije `setup()` i `draw()`
- Tok programa
- Interakcija s mišem

Tok programa

Šta je blok koda?

Blok koda je bilo koji kod koji se nalazi untar vitičastih zagrada.

```
{
        Neki kod
}
```

Funkcije setup() i draw()

Sintaksa:

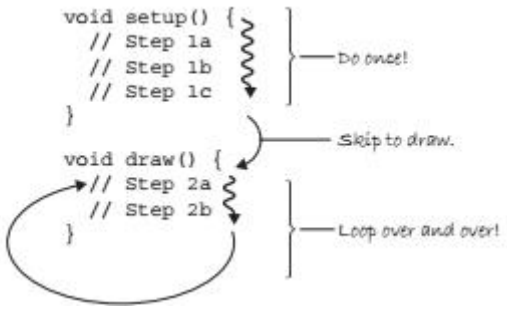
```
void setup() {
        //Kod za inicijalizaciju
}

void draw() {
        //Kod koji se obavlja beskonačno dugo.
}
```

Tok programa

Funkcija setup() se obavlja samo jednom, odmah po pokretanju programa. Ona nam služi za definiranje početnih vrijednosti. Prva linija koda funkcije setup() je uvijek funkcija size(), često praćena kodom za definiranje boja (fill i stroke) ili možda učitavanja slika i fontova.

Kod unutar funkcije draw() izvršava se od vrha ka dnu a zatim se ponavlja sve dok ne kliknemo dugme za prekid programa (Stop) ili zatvorimo prozor. Svaki prolaz kroz funkciju draw() naziva se *frame* (engl. Okvir). Postavljena brzina promjene okvira je 60 okvira po sekundi, no to se može i promijeniti.



1 Tok programa

Vježba br. 1

Nacrtajmo Zooga ☺

```
void setup(){
// Postavi veličinu prozora
size(200,200);
}
void draw() {
// Bijela pozadina
background(255);
// Postavi mod crtanja
ellipseMode(CENTER);
rectMode(CENTER);
// Zoogovo tijelo
stroke(0);
fill(150);
rect(100,100,20,100);
// Zoogova glava
stroke(0);
fill(255);
ellipse(100,70,60,60);
// Zoogove oči
fill(0);
ellipse(81,70,16,32);
ellipse(119,70,16,32);
// Nacrtaj Zoogove noge
stroke(0);
line(90,150,80,160);
line(110,150,120,160);
}
```

Vježba br. 2

Funkcija println()

Ova funkcija nam služi za ispis teksta.

Testirajte kod:

```
void setup() {
println("I'm starting");
}
void draw() {
println("I'm running");
}
```

Interakcija s mišem

Šta ako bismo umjesto unosa broja u neku od funkcija za crtanje mogli unijeti lokaciju miša na prozoru (x i/ili y koordinatu)? Na primjer:

```
line(x koordinata lokacije miša, y koordinata lokacije miša, 100, 100);
```

Ovo zaista možemo uraditi, s tim da naravno bismo morali koristiti određene ključne riječi za koordinate miša. To su: **`mouseX`** i **`mouseY`**.

Primjer 2:

```
void setup() {
size(200,200);
}
void draw() {
background(255);

stroke(0);
fill(175);
rectMode(CENTER);
rect(mouseX,mouseY,50,50);
}
```

Vježba br.3

Nacrtajte krug prečnika 60 piksela čiji je centar trenutna lokacija miša.

Vježba br.4

Ukoliko bismo u primjeru 2 premjestili background() komandu iz funkcije draw() i premjestili je u setup() dobili bismo sljedeću sliku:

Zašto se ovo desilo?

Vježba br.5

U ovom primjeru želimo da se tijelo i glava našeg Zooga pokreću, dok oči i noge ostaju nepomične.

```
void setup() {
size(200,200);
smooth();
}

void draw() {
background(255);
ellipseMode(CENTER);
rectMode(CENTER);

// Nacrtajmo tijelo
stroke(0);
fill(175);
rect(mouseX,mouseY,20,100);

// Nacrtajmo glavu
stroke(0);
fill(255);
ellipse(mouseX,mouseY-30,60,60);

// Oči
fill(0);
ellipse(81,70,16,32);
ellipse(119,70,16,32);

// Noge
stroke(0);
```

```
line(90,150,80,160);
line(110,150,120,160);
}
```

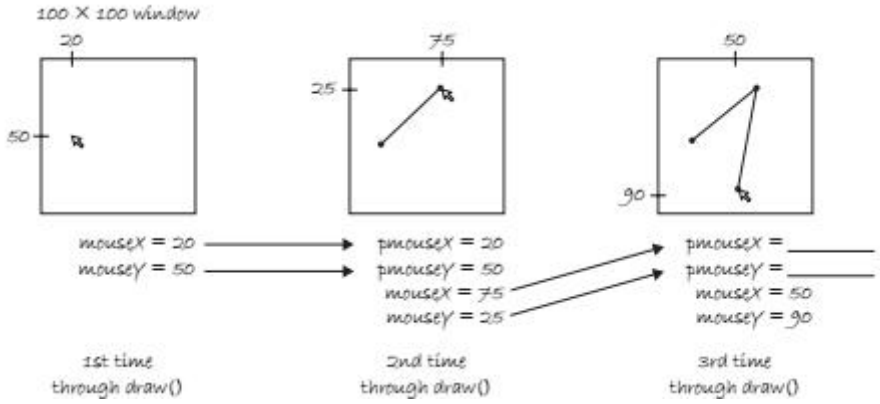
Vježba br.6

Kako da modificiramo gornji kod da i ostatak Zooga (oči i noge) prate miš?

Vježba br.7

Modificirajte kod domaćeg zadatka s prethodnih vježbi tako da karakter koji ste kreirali prati miš.

Pored **mouseX** i **mouseY** postoje još dvije ključne riječi koje možemo koristiti: **pmouseX** i **pmouseY**. Ove dvije ključne riječi predstavljaju prethodne koordinate lokacije miša, tj, gdje je miš bio zadnji put kada smo prošli kroz draw() funkciju. Ovo nam pruža nekoliko interesantnih mogućnosti za interakciju s mišem. Na primjer, razmotrimo šta se dešava ukoliko povučemo liniju od prethodne lokacije miša do trenutne.



Povezivanjem prethodne lokacije miša s trenutnom svaki put kada prođemo kroz draw(9 u mogućnosti smo da iscrtamo neprekidnu liniju koja prati miš.

Vježba br.8

Crtanje neprekidne linije.

```
void setup() {
size(200,200);
background(255);
smooth();
}
void draw() {
stroke(0);
line(pmouseX,pmouseY,mouseX,mouseY);
}
```

pmouseX i **pmouseY** mogu se koristiti također i za proračun brzine miša. Ovo se postiže računanjem distance između trenutne i prethodne pozicije miša. Ukoliko se miš kreće sporo, distanca

je mala, no ukoliko se miš počne kretati brže, distanca se povećava. Distancu računamo kao apsolutnu razliku između trenutne i prethodne lokacije, no funkcija dist() nam olakšava ovaj proračun.

Vježba br.9

U ovom primjeru postavljamo debljinu poteza prema brzini pomjeranja miša.

```
void setup() {
size(480, 120);
smooth();
stroke(0, 102);
}
void draw() {
float weight = dist(mouseX, mouseY, pmouseX, pmouseY);
strokeWeight(weight);
line(mouseX, mouseY, pmouseX, pmouseY);
}
```