

Звіт

з обчислюваної геометрії
та комп'ютерної графіки

Ганевича Вадіма

12.04.2022

Умова:

Центр кола рухається на відрізок

$$x = x_0 + lt, \quad y = y_0 + mt$$

Точка на колі рухається з частотою

$$\omega \quad (x = x_c + R \cos \omega t, \quad y = y_c + R \sin \omega t)$$

Зобразити графік, який отримає точка

на колі. Параметр $t \in [0, 6\pi]$

Математичний розв'язок:

Для вирішення поставленої задачі, ми використовуємо формули знаходження координат центра кола, які залежать від параметра t :

$$x = x_0 + l t, \quad y = y_0 + m t$$

Також, щоб знайти координати точки на колі, яка також залежить від параметра t , ми використовуємо наступні формули:

$$x = x_c + R \cos \omega t, \quad y = y_c + R \sin \omega t$$

Алгоритм розв'язку:

Крок 1:

Ми маємо два різновиди об'єктів, які будемо обробляти - коло та точка.

Для коректного руху ці об'єкти повинні містити методи, які відображають об'єкт на екрані, витирають його звідти та обновляють координати.

Функцію, яка витирає робочу область містить бібліотека, з якою я працюю, тому її тут не реалізовано.

Створимо такі об'єкти.

```
class Circle{
    constructor(center, r) {
        this.center = center;
        this.r = r;
    }

    updateCenter(start, t){
        this.center.x = start.x + 20 * t;
        this.center.y = start.y + 20 * t;
    }

    show(stage){
        stage.circle(this.center.x, this.center.y, this.r);
        stage.circle(this.center.x, this.center.y, 1);
    }
}
```

```
class Point{
    constructor(radius, color) {
        this.r = radius;
        this.color = color;
        this.x = this.r * Math.cos( x: 0);
        this.y = this.r * Math.sin( x: 0);
    }

    updatePoint(center, t){
        this.x = center.x + this.r * Math.cos( x: 4 * t);
        this.y = center.y + this.r * Math.sin( x: 4 * t);
    }

    show(stage){
        stage.circle(this.x, this.y, 1)
            .stroke(this.color);
    }
}
```

Крок 2:

Коли ми створили безпосередні зразки об'єктів, з якими працюватимемо - нам потрібно створити об'єкт робочої області, на яку ми малюватимемо ці об'єкти.

Вона керуватиме рухом та управляти усіма об'єктами, що бачить користувач.

```
class Frame{
  constructor() {
    this.stage = acgraph.create('stage-container');
    this.layer = this.stage.layer();
    this.circle = new Circle();
    this.point = new Point();
  }

  #show(){
    this.circle.show(this.layer);
    this.point.show(this.layer);
  }

  #clear(){
    this.layer.removeChildren();
  }
}
```

Крок 3:

Все необхідне для роботи створено.

На цьому кроці потрібно об'єднати все, що було написано як окремі об'єкти, в одну концепцію, яка рухається.

Для цього використаємо цикл для параметра t:

```
async move(startPos){
  this.circle = new Circle( center: {...startPos}, r: 40);
  this.point = new Point( radius: 40, color: "red");
  for (let t = 0; t < 6 * Math.PI; t += 0.01){
    this.#clear();
    this.#show();
    this.circle.updateCenter(startPos, t);
    this.point.updatePoint(this.circle.center, t);

    let p = new Point( radius: 40, color: "blue");
    p.updatePoint(this.circle.center, t);
    p.show(this.stage);

    // sleep
    await new Promise( executor: resolve => setTimeout(resolve, timeout: 20));
  }
}
```

Рух відбувається таким чином:

- 1. Витираються усі фігури**
- 2. Обновляються координати цих фігур**
- 3. Виводяться фігури з оновленими координатами**
- 4. Повторення, поки певна умова не буде задовільнена**

Приклади роботи програми:



