SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로! Enabling People

9기 자기주도 학습

간단한 스네이크 게임 구현

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People
Education for Future Generations

간단한 스네이크 게임 구현

이 현 석 Project consultant

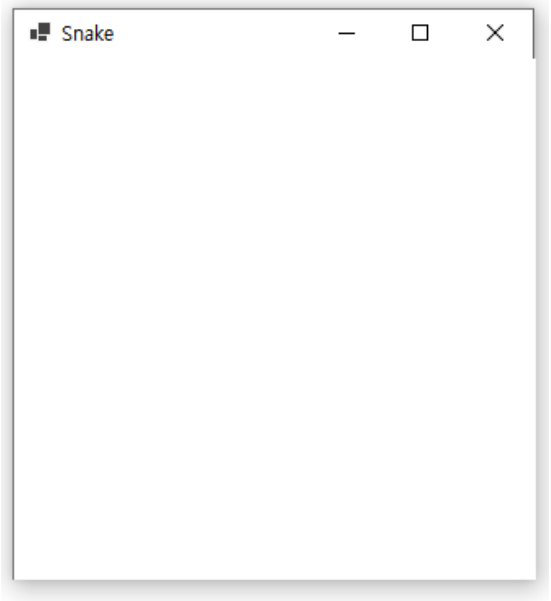
- 삼성 물산 POS program 및 8’ kiosk 제작
- 인디게임 제작 업체 H2S 대표이사
- 4:33 모로저택의 비밀 앱 메인 프로그래머
- LG Embeded 게임 주브레이커 앱 및 엔진 제작
- 광운대학교 컴퓨터 공학 학사

스크린 창 설정

참고 내용

- C# windows application 기준으로 구현
- 가로 크기 / 세로 크기 지정
- 윈도우 Size 생성

```
// 전체 해상도
static int m_mapX = 400;
static int m_mapY = 400;

참조 1개
public SnakeGame()
{
    InitializeComponent();

    // 더블 버퍼링 적용
    SetStyle(ControlStyles.OptimizedDoubleBuffer, true);

    // 윈도우 영역 세팅
    this.ClientSize = new Size(m_mapX, m_mapY);
}
```

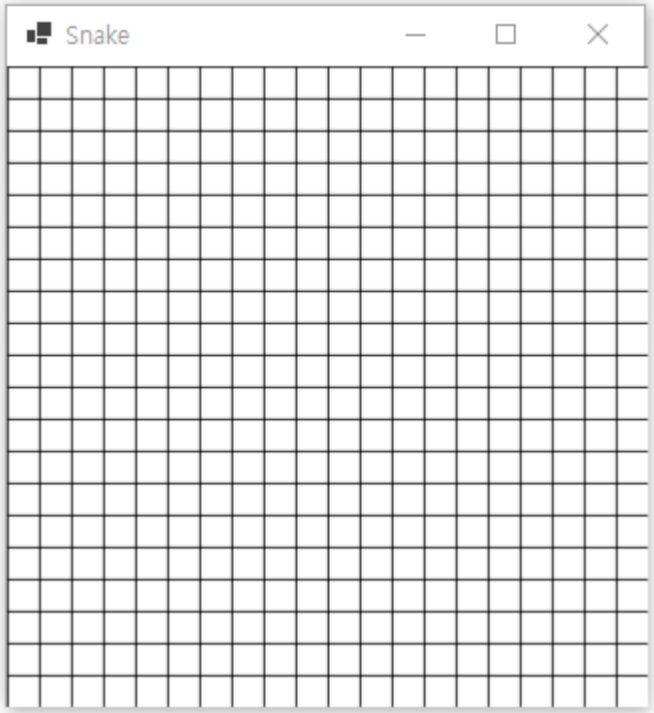


맵 그리기

참고 내용

- 가로 세로에 그려야 할 선의 개수 구함
- Pen 객체를 통해 가로선을 20pixel 씩 20개 그림
- Pen 객체를 통해 세로선을 20pixel 씩 20개 그림

```
//선 그리기
참조 1개
private void DrawMap(PaintEventArgs e)
{
    //간격으로 선그린다.
    int widthNum = m_mapX / m_objectX;
    int heightNum = m_mapY / m_objectY;

    //가로선 그리기
    Pen drawPen = new Pen(Color.Black, 1);
    Point startX = new Point();
    Point endY = new Point();

    startX.X = 0;
    startX.Y = 0;
    endY.X = m_mapX;
    endY.Y = 0;

    for (int j = 0; j < widthNum; j++)
    {
        e.Graphics.DrawLine(drawPen, startX, endY);
        startX.Y += m_objectY;
        endY.Y += m_objectY;
    }

    //세로선 그리기
    startX.X = 0;
    startX.Y = 0;
    endY.X = 0;
    endY.Y = m_mapY;

    for (int i = 0; i < heightNum; i++)
    {
        e.Graphics.DrawLine(drawPen, startX, endY);
        startX.X += m_objectX;
        endY.X += m_objectX;
    }
}
```



뱀 머리, 먹이 설정

참고 내용

- 스네이크 객체 / 먹이 dass 디자인
- 스네이크 객체는 Ratangle 객체를 통해 x, y위치, 너비, 높이를 표현
- 스네이크 객체의 방향은 Up / Down / Left / Right 4가지로 설정
- 제공되는 함수는 Copy(), Update(), Draw(), CheckColision()으로 구현
- 먹이는 스네이크 객체를 상속받아 구현
- 추가되는 함수는 Create()이고, 먹이 위치를 재설정하는 함수
  (단 뱀의 위치와 겹치지 않는 valiation check 필요)

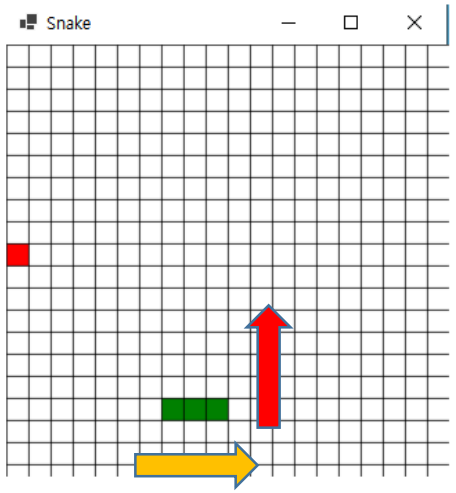
```
// 스네이크 몸통 클래스
참조 25개
public class SnakeBody
{
    // 그리는 영역
    public Rectangle rt = new Rectangle();
    // 그리는 색
    public Brush brush = new SolidBrush(Color.Green);
    // 생성자
    참조 4개
    public SnakeBody()
    {
        rt.Width = m_objectX;
        rt.Height = m_objectY;
    }
    // 방향 (머리일때만 사용)
    public int m_direction = D_RIGHT;
    // 데이터 복사
    참조 1개
    public void Copy(SnakeBody nextBody)
    {
        rt = nextBody.rt;
        m_direction = nextBody.m_direction;
    }
    // 업데이트
    참조 1개
    public void Update()...
    // 뱀 그리기
    참조 2개
    public void Draw(PaintEventArgs e)...
    // 충돌 체크
    참조 2개
    public bool CheckCollision(SnakeBody someObject)...
}
```

```
// 사과 클래스
참조 3개
public class Apple : SnakeBody
{
    참조 1개
    public Apple()
    {
        brush = new SolidBrush(Color.Red);
    }

    // 랜덤 생성
    참조 2개
    public void Create()
    {
        // 뱀의 위치와 중복되지 않게 생성한다.
        bool isAppleCraeted = false;
        while(isAppleCraeted == false)
        {
            Random rand = new Random();
            rt.X = rand.Next(0, m_mapX / m_objectX) * m_objectX;
            rt.Y = rand.Next(0, m_mapY / m_objectY) * m_objectY;

            // 현재 뱀의 위치와 같은지 체크한다.
            bool isSamePosition = false;
            for(int i=0; i< m_snakeList.Count;i++)
            {
                SnakeBody snakeBody = (SnakeBody)m_snakeList[i];
                if(rt == snakeBody.rt)
                {
                    isSamePosition = true;
                    break;
                }
            }
            if (isSamePosition == true)
                continue ;
            isAppleCraeted = true;
        }

    }
};
```

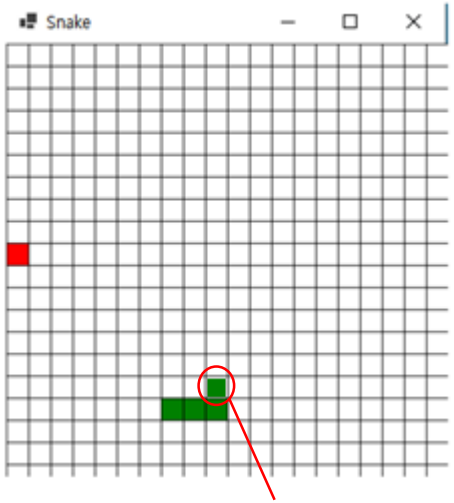
뱀의 이동 로직 구현

참고 내용

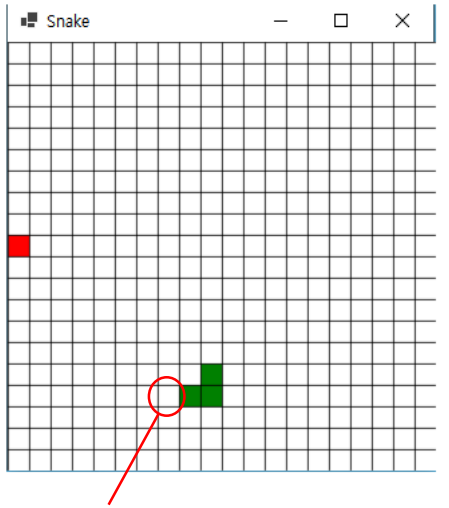
- 뱀 이동 로직은 뱀 머리가 4방향 중 한 곳을 이동 한 뒤, 뱀 리스트에 추가한다.
- 이동한 곳에 뱀의 먹이가 존재하지 않으면 가장 마지막 리스트 데이터를 삭제한다.
- 뱀의 먹이가 존재하면 리스트 데이터를 삭제하지 않는다.
- 게임 오버 조건을 체크한다.



뱀의 머리가 오른쪽 -> 위쪽으로 바뀜



위쪽 방향으로 뱀의 머리를 추가한다.



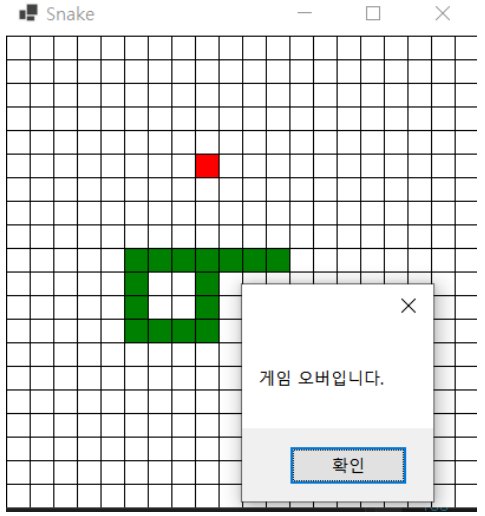
먹이가 없으므로 가장 마지막 데이터를 지운다.

뱀의 이동 로직 구현

참고 내용

- 뱀의 머리가 움직인 곳이 화면 밖 (뱀의 머리위치.X < 0 or 뱀의 머리위치.X >= 400 or 뱀의 머리위치.Y < 0 or 뱀의 머리위치.Y >= 400)에 존재하면 게임 오버를 시킨다.
- 뱀의 머리가 움직인 곳이 뱀 리스트의 X, Y와 같은 경우 게임 오버를 시킨다.

뱀의 머리가 화면 밖을 나간 경우



뱀의 머리가 자신의 몸통에 닿은 경우

해설 요약

참고 내용

- 스네이크 게임의 기본 구성은 전체 맵 그리기, 스네이크 이동 로직, 먹이 존재 여부, 게임 오버 조건 체크로 나눌 수 있다
- 전체 맵을 그리기 위해서는 GUI 객체로 가로 / 세로 선 개수 만큼 그려주면 된다.
- 스네이크 객체를 구현할 때는 이동 위치 및 방향을 세팅할 수 있어야 한다.
- 먹이 객체는 스네이크 객체를 상속받아 구현하면 편리하다.
- 스네이크의 이동 구현 핵심은 머리의 이동 방향으로 리스트 앞에 추가하고, 먹이 유무에 따라 리스트의 마지막 부분을 처리하는 것이다.
- 게임 오버 조건은 뱀의 머리가 몸통에 닿거나 화면 밖으로 나가는 경우를 체크한다.