App development for everyone

Building a Breakout clone with javascript

Lesson 4

- Homework
- Keep info together
- More lists

This loop always draws 54 bricks

```
for(let row=0; row<6; row++) {
  for(let col=0; col<9; col++) {
    fill(brickColors[row]);
    rect(col * brickWidth + col * brickMargin + borderWidth + brickMargin,
       row * brickHeight + row * brickMargin + borderWidth + brickMargin, brickWidth,
       brickHeight);
  }
}
```

How do we remove bricks?

- Keep bricks in a list
- Remove from list when a brick is hit

```
let bricks = [];
```

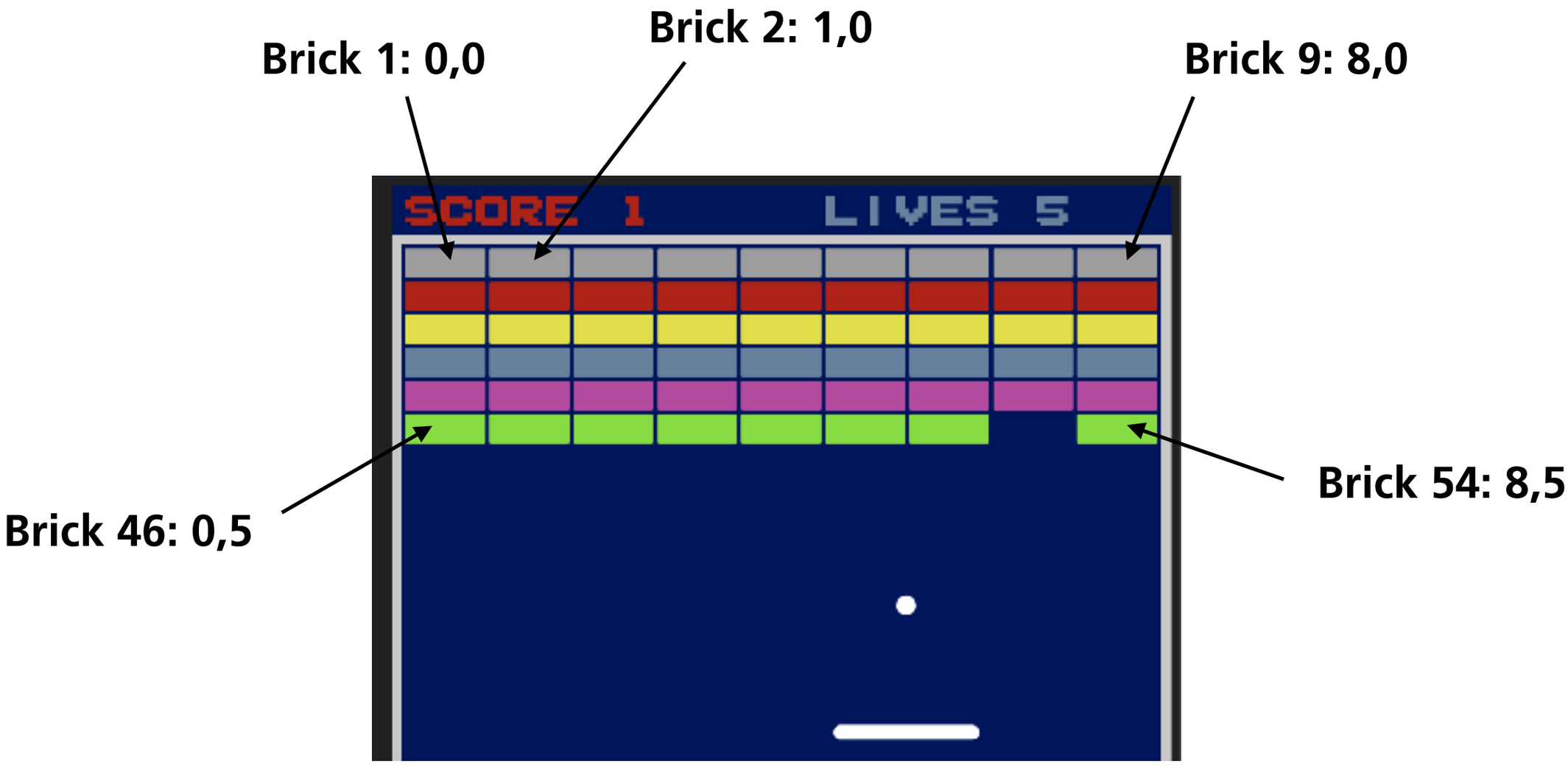
Add bricks to list

- In javascript, you use push to add an item to a list

```
for(let row=0; row<6; row++) {
  for(let col=0; col<9; col++) {
    bricks.push( ??? );
  }
}
```

- But what do we push?

Keep info together: x and y



Javascript object notation

- Use { and } to keep info together

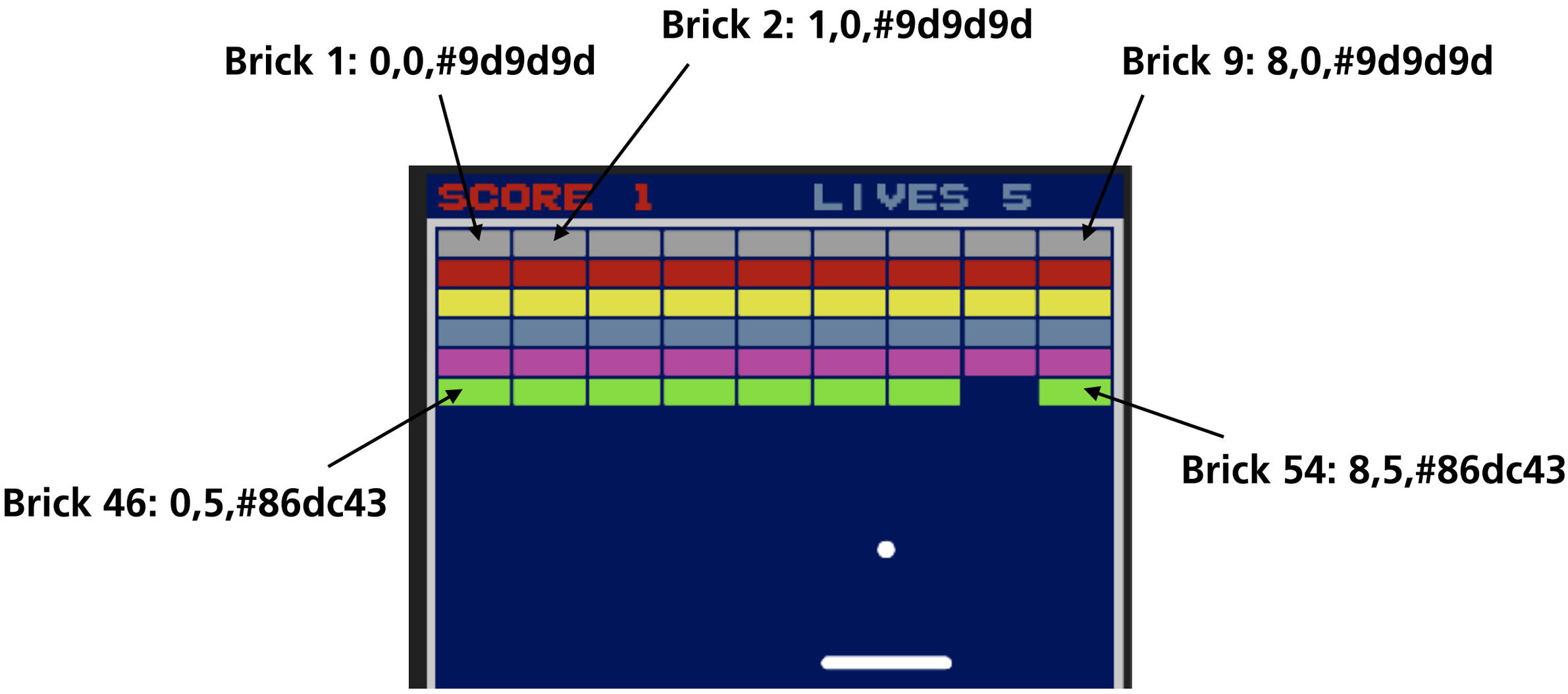
```
> brick = { x: 10, y: 20 }
< ▶ {x: 10, y: 20}
> brick
< ▶ {x: 10, y: 20}
> brick.x
< 10
>
```

Brick with x and y

```
for(let row=0; row<6; row++) {
  for(let col=0; col<9; col++) {
    bricks.push({
      x: col * brickWidth + col * brickMargin + borderWidth + brickMargin,
      y: row * brickHeight + row * brickMargin + borderWidth + brickMargin,
    });
  }
}
```

```
▶ 0: {x: 14, y: 14}
▶ 1: {x: 100, y: 14}
▶ 2: {x: 186, y: 14}
▶ 3: {x: 272, y: 14}
▶ 4: {x: 358, y: 14}
▶ 5: {x: 444, y: 14}
▶ 6: {x: 530, y: 14}
▶ 7: {x: 616, y: 14}
▶ 8: {x: 702, y: 14}
▶ 9: {x: 14, y: 48}
▶ 10: {x: 100, y: 48}
▶ 11: {x: 186, y: 48}
▶ 12: {x: 272, y: 48}
▶ 13: {x: 358, y: 48}
▶ 14: {x: 444, y: 48}
▶ 15: {x: 530, y: 48}
▶ 16: {x: 616, y: 48}
▶ 17: {x: 702, y: 48}
```

Add color to brick



Add color

```
function setup() {
  createCanvas(gameWidth, gameHeight);
  setupRound();
}

function setupRound() {
  for(let row=0; row<6; row++) {
    for(let col=0; col<9; col++) {
      bricks.push({
        x: col * brickWidth + col * brickMargin + borderWidth + brickMargin,
        y: row * brickHeight + row * brickMargin + borderWidth + brickMargin,
        color: brickColors[row],
      });
    }
  }
}
```

```
▶ 0: {x: 14, y: 14, color: "#9d9d9d"}
▶ 1: {x: 100, y: 14, color: "#9d9d9d"}
▶ 2: {x: 186, y: 14, color: "#9d9d9d"}
▶ 3: {x: 272, y: 14, color: "#9d9d9d"}
▶ 4: {x: 358, y: 14, color: "#9d9d9d"}
▶ 5: {x: 444, y: 14, color: "#9d9d9d"}
▶ 6: {x: 530, y: 14, color: "#9d9d9d"}
▶ 7: {x: 616, y: 14, color: "#9d9d9d"}
▶ 8: {x: 702, y: 14, color: "#9d9d9d"}
▶ 9: {x: 14, y: 48, color: "#ae2317"}
▶ 10: {x: 100, y: 48, color: "#ae2317"}
▶ 11: {x: 186, y: 48, color: "#ae2317"}
▶ 12: {x: 272, y: 48, color: "#ae2317"}
▶ 13: {x: 358, y: 48, color: "#ae2317"}
▶ 14: {x: 444, y: 48, color: "#ae2317"}
```

Flat list

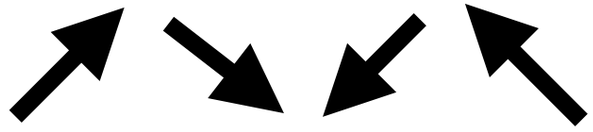
- We now have a list of 54 bricks.
- Each brick has x, y and color.
- Use forEach to draw the bricks.

```
bricks.forEach(brick => {
  fill(brick.color);
  rect(brick.x, brick.y, brickWidth, brickHeight);
});
```

Homework

- We now know that a brick has x, y and color. We have learned how to keep this information together.
- The ball also has some information. Design a ball-object

Ball

- The ball has a variable position (x and y)
- The ball has a fixed radius
- The ball has a variable direction 
- How does the javascript object look like?

```
ball = {
  ???
  ???
  ???
};
```

hauptstadtcoder@fastmail.com