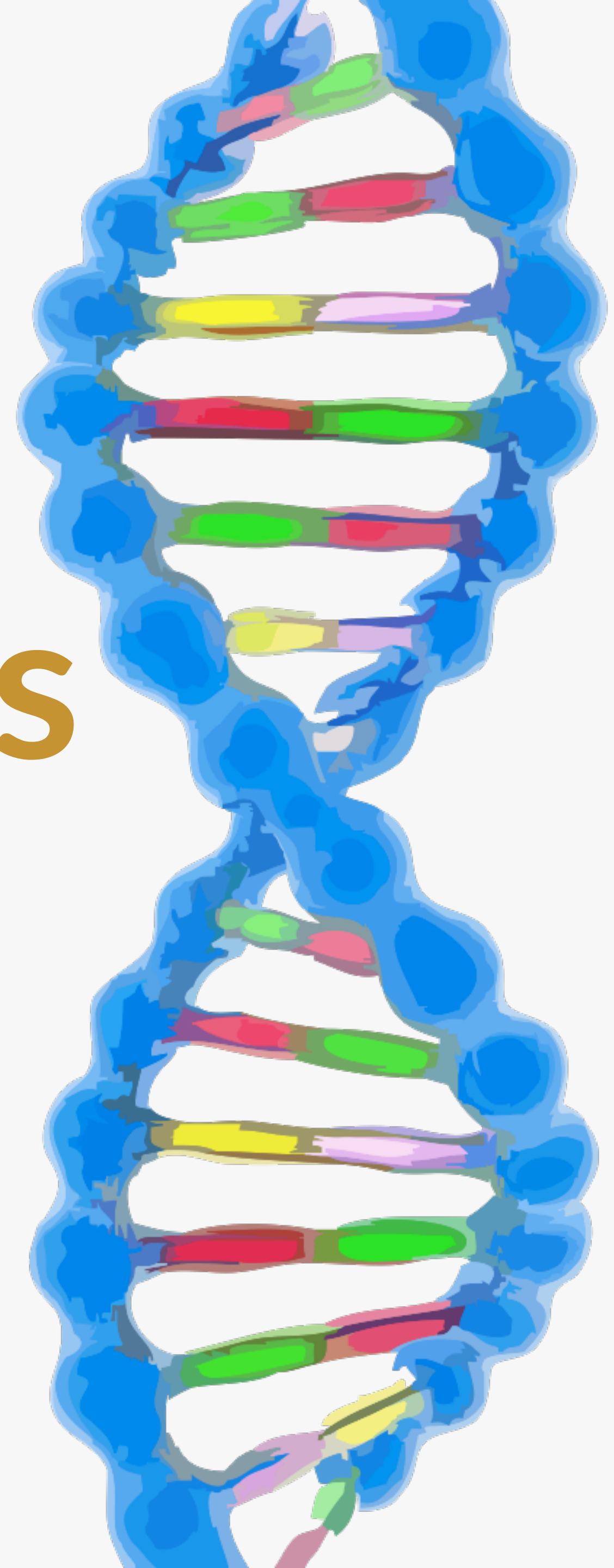


# Evolving Game Development With **Genetic Algorithms**

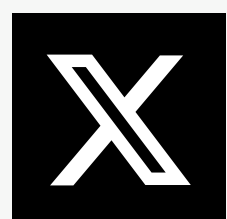
## Здраво

Kevin Maes | What The Stack - September 14, 2024





**Kevin Maes**



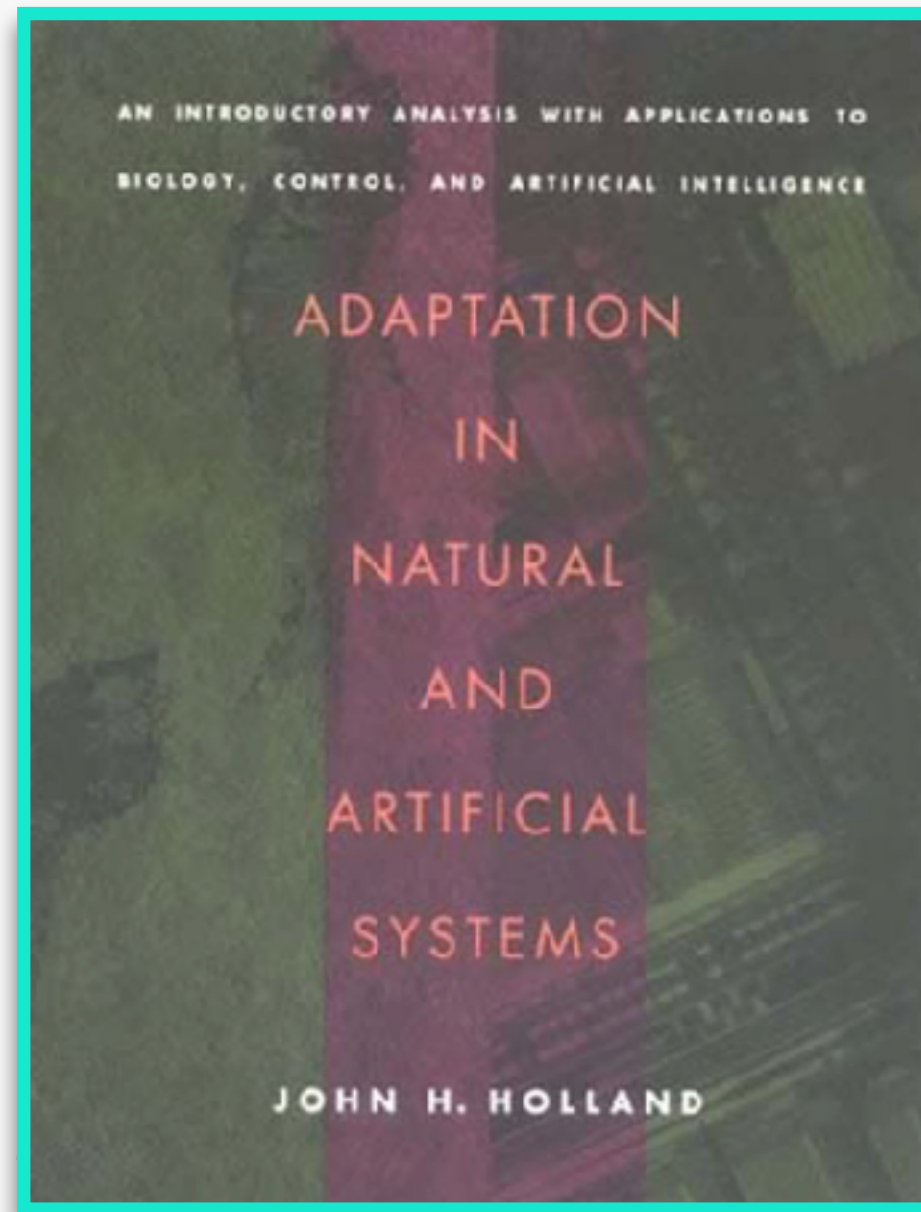
**STATELY**





# Genetic Algorithms

## Adaptation in Natural and Artificial Systems



*“general theories of adaptive processes apply across biological, cognitive, social, and computational systems”*



John H. Holland

# Why genetic algorithms?



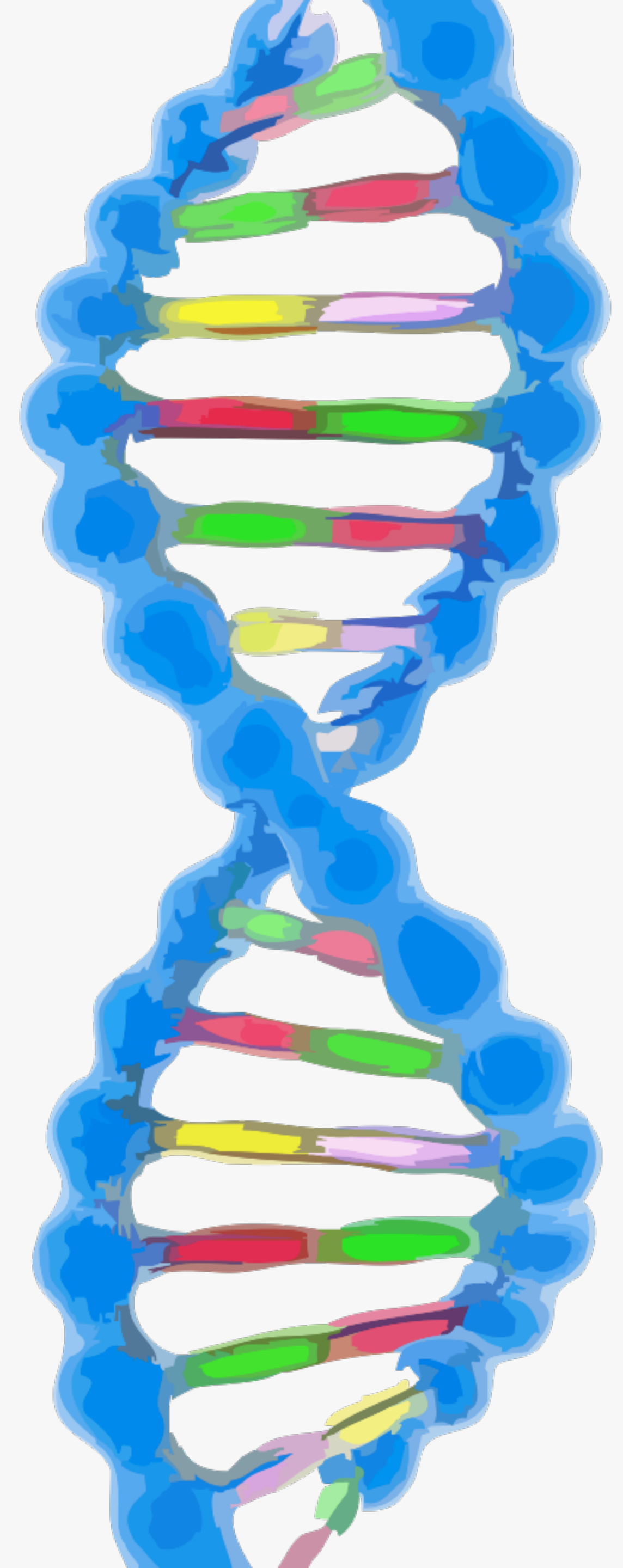
Desired behavior



Optimal design

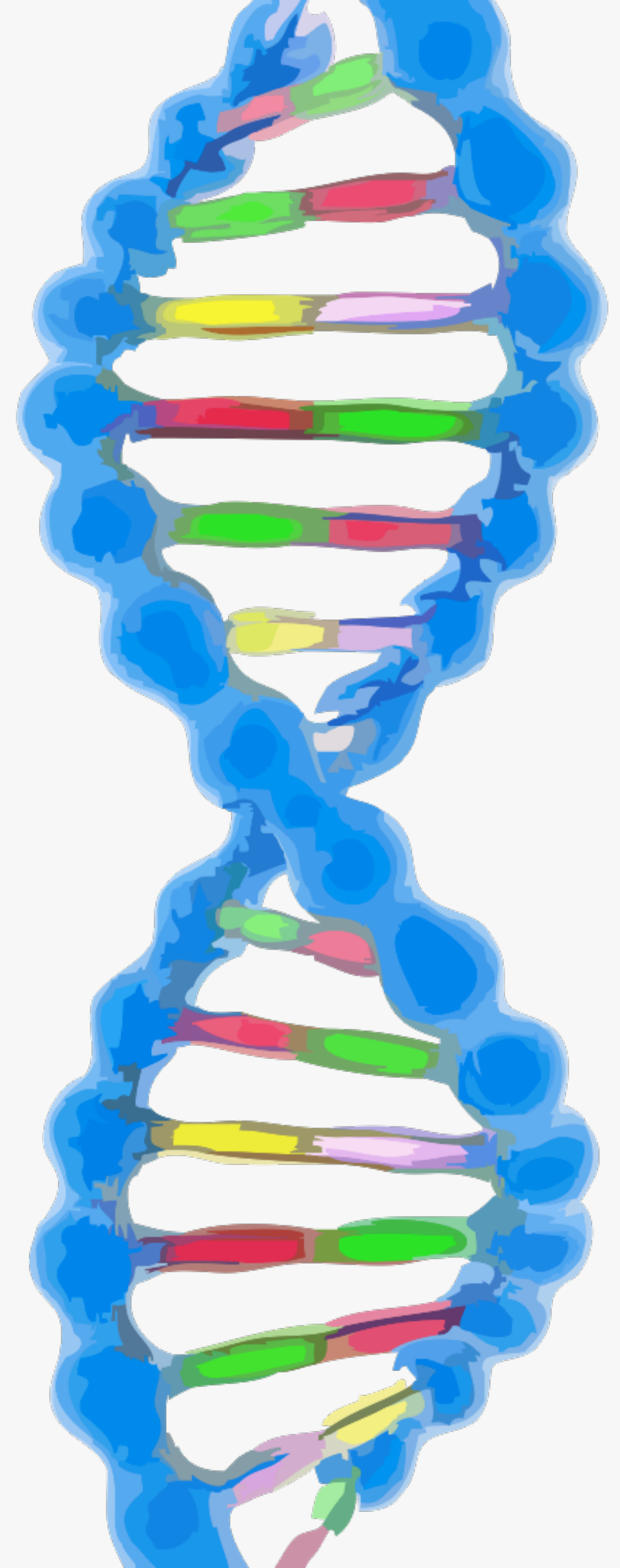


Solve complex search problems

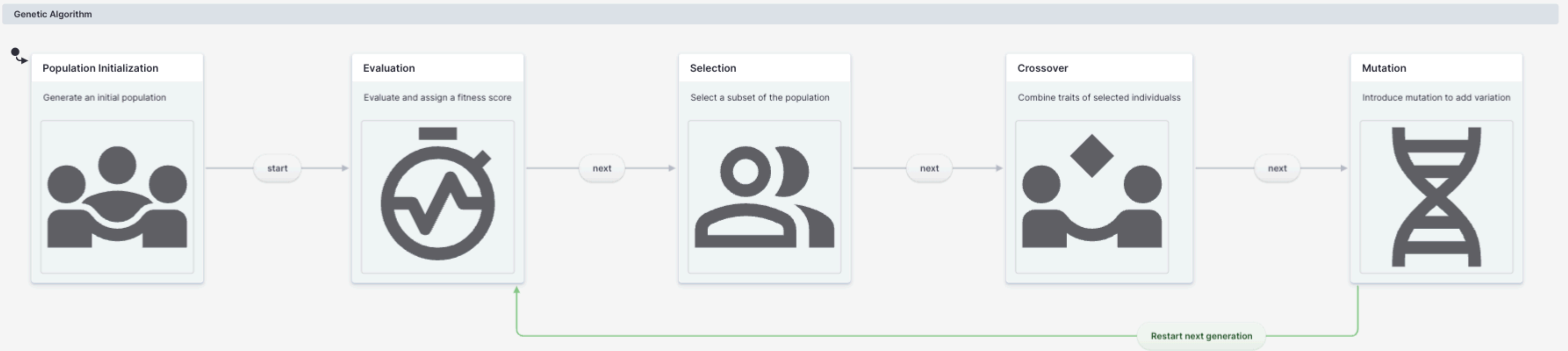


# There's a GA for that!

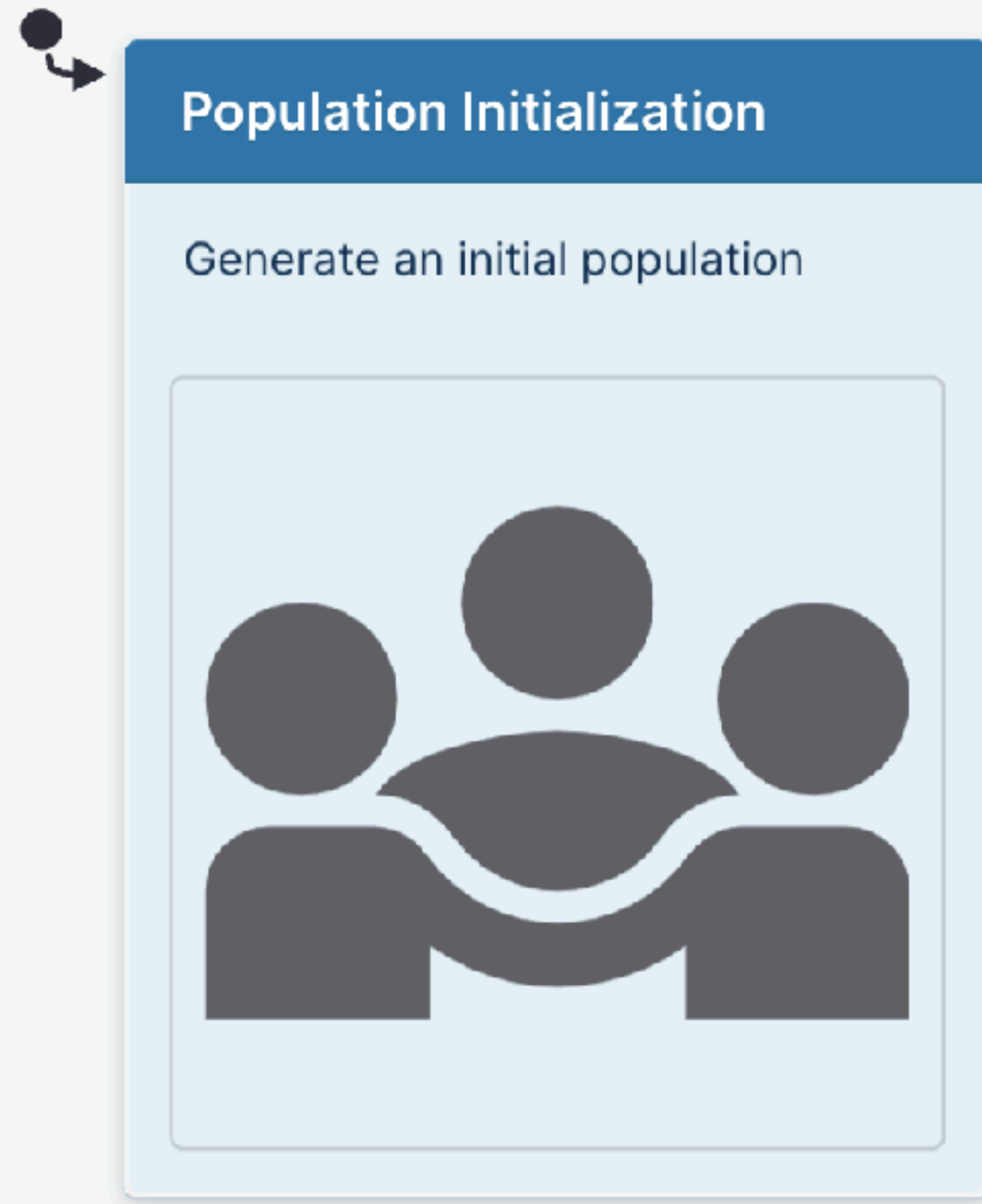
- Designing efficient network topologies
- Automated software testing
- Evolving game strategies
- Scheduling Problems
- Complex hardware design



# States of a genetic algorithm



# Population Initialization



Many “individuals”



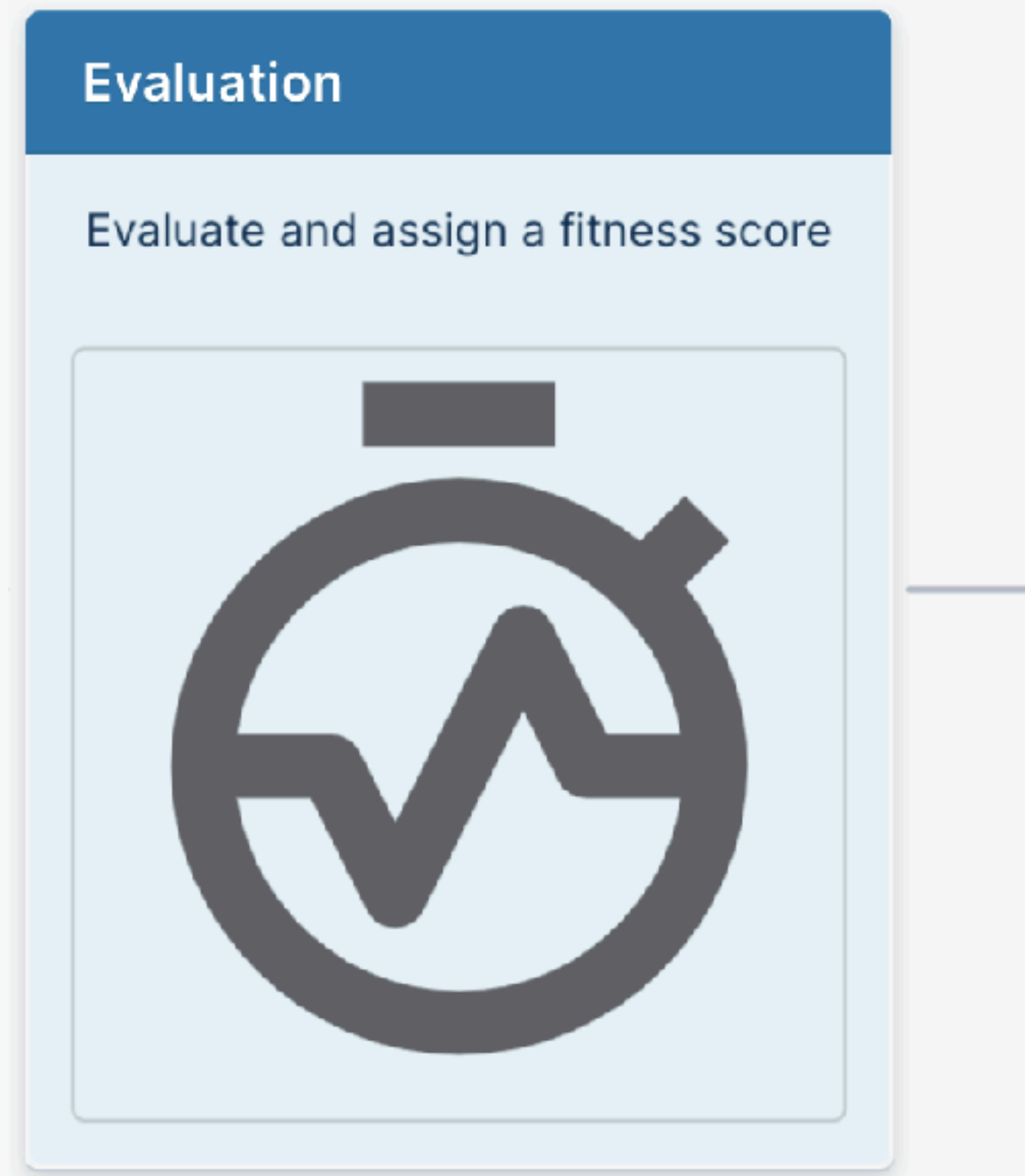
Each with a potential solution





Stored in their “DNA”



# Evaluation & Fitness



-  Evaluate performance
-  Reward behavior
-  Punishment
-  Weighted criteria

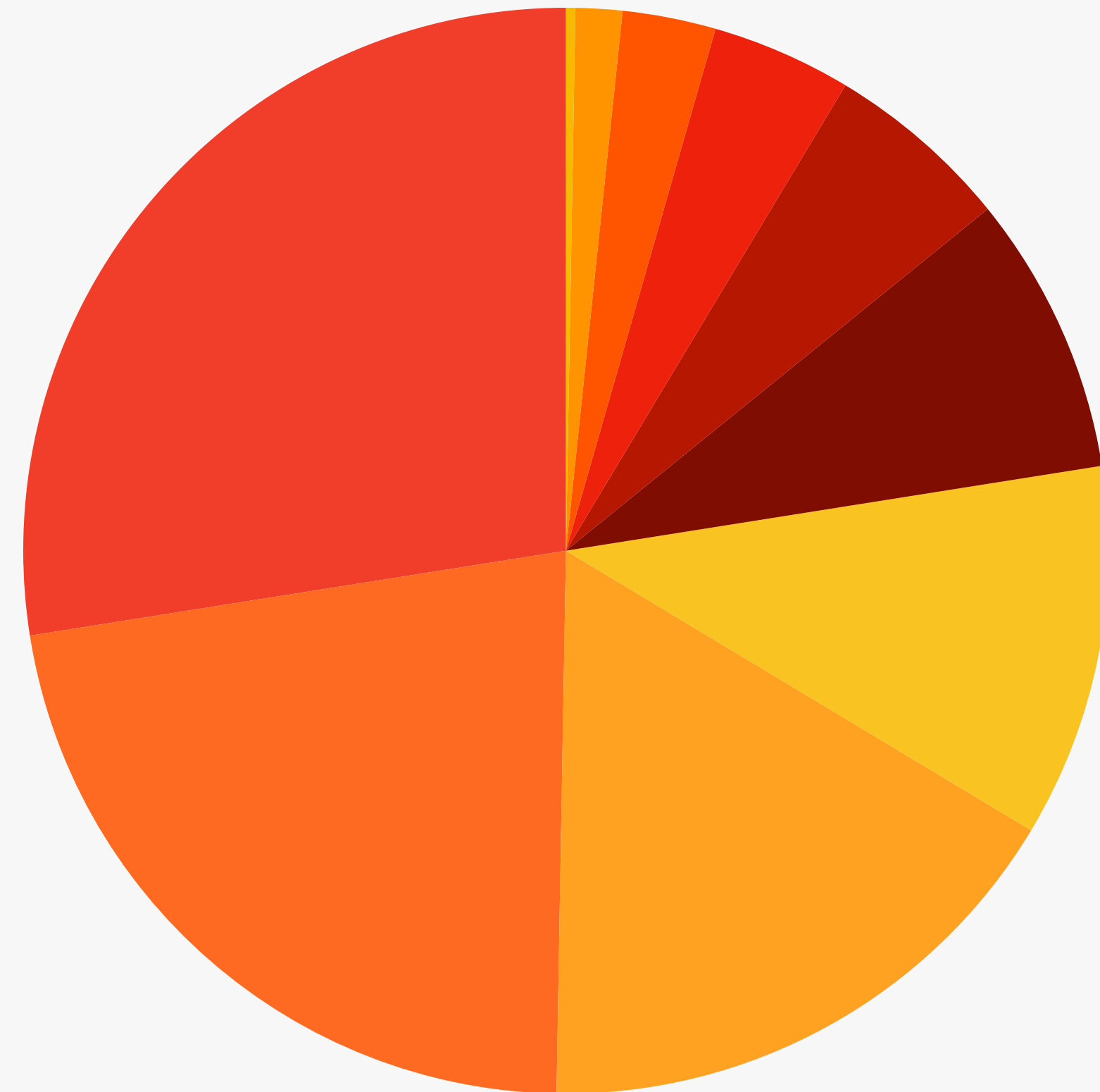
# Selection

**Selection**

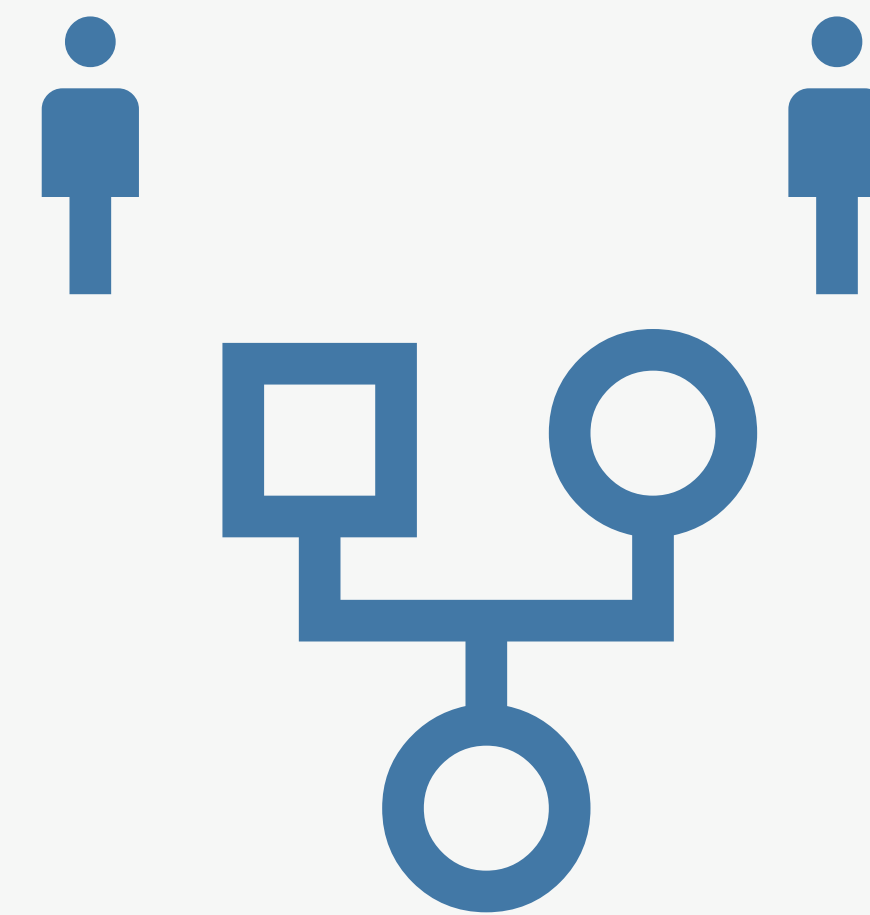
Select a subset of the population

A light blue rectangular box with a dark blue header containing the word "Selection" in white. Below the header, the text "Select a subset of the population" is displayed. At the bottom of the box is a dark gray icon of two stylized human figures, one slightly behind the other, representing a population.

## Roulette Wheel Selection

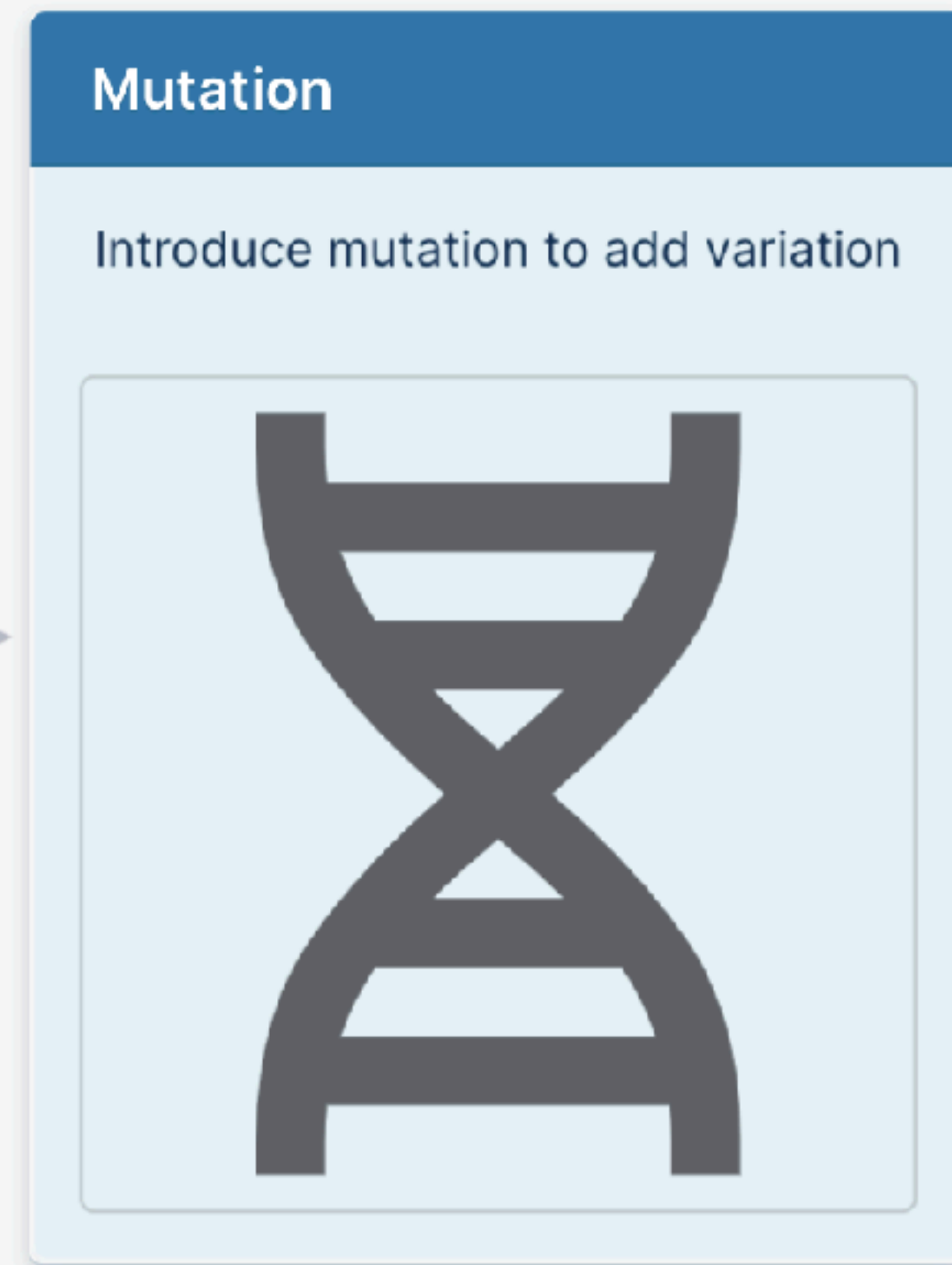


# Crossover



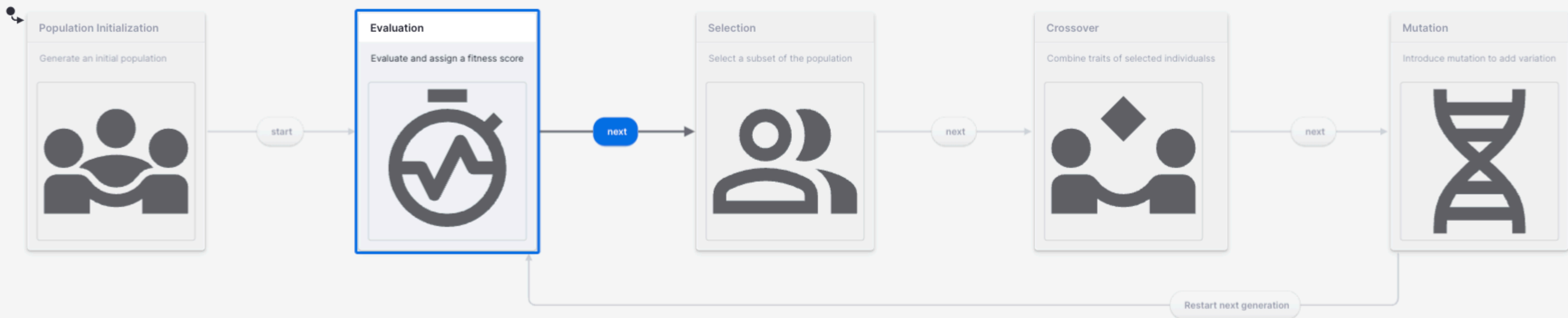
 x total population size

# Mutation



- Need to maintain variation
- Avoid “local optima”
- Strive for the “global optimum”
- Mutation rate
- Mutation amount

Genetic Algorithm







# MUSEO VIDEOJUEGO MÁLAGA

MUSEO VIDEOJUEGO MÁLAGA OXO

PLAZA DEL SIGLO

anish School

OExpo

Escuela de español

nicaja

OXO STORE

OXO STORE

OXO

[oxomuseo.com](http://oxomuseo.com)

# Museo Videojuego Malaga



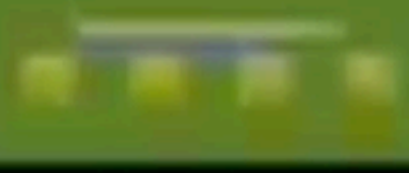
**Atari 2600**



**Nintendo**



616









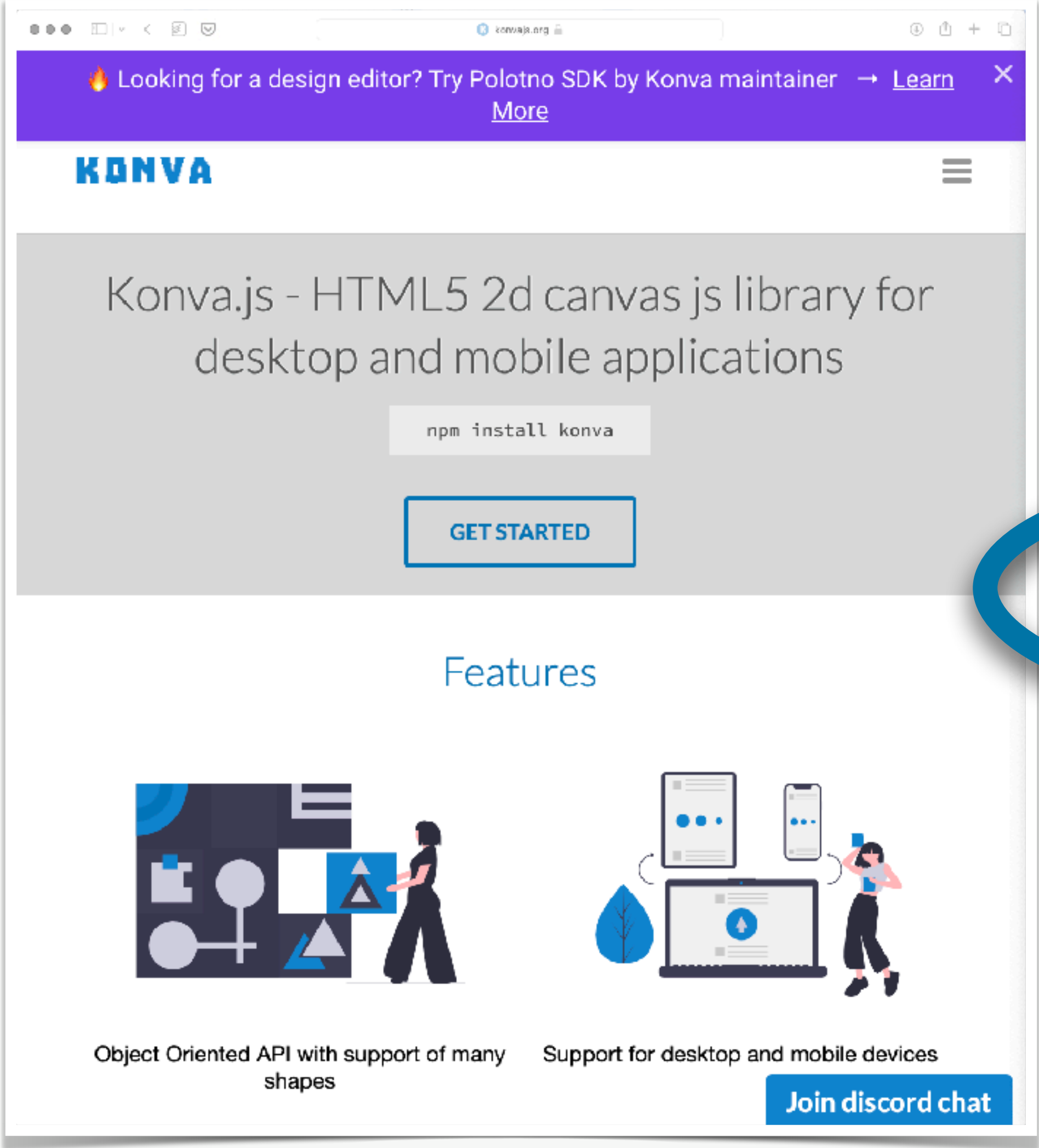
# Egg drop



# Egg drop



# React + Konva



Looking for a design editor? Try Polotno SDK by Konva maintainer → [Learn](#) ✕  
[More](#)

**KONVA**

Konva.js - HTML5 2d canvas js library for desktop and mobile applications

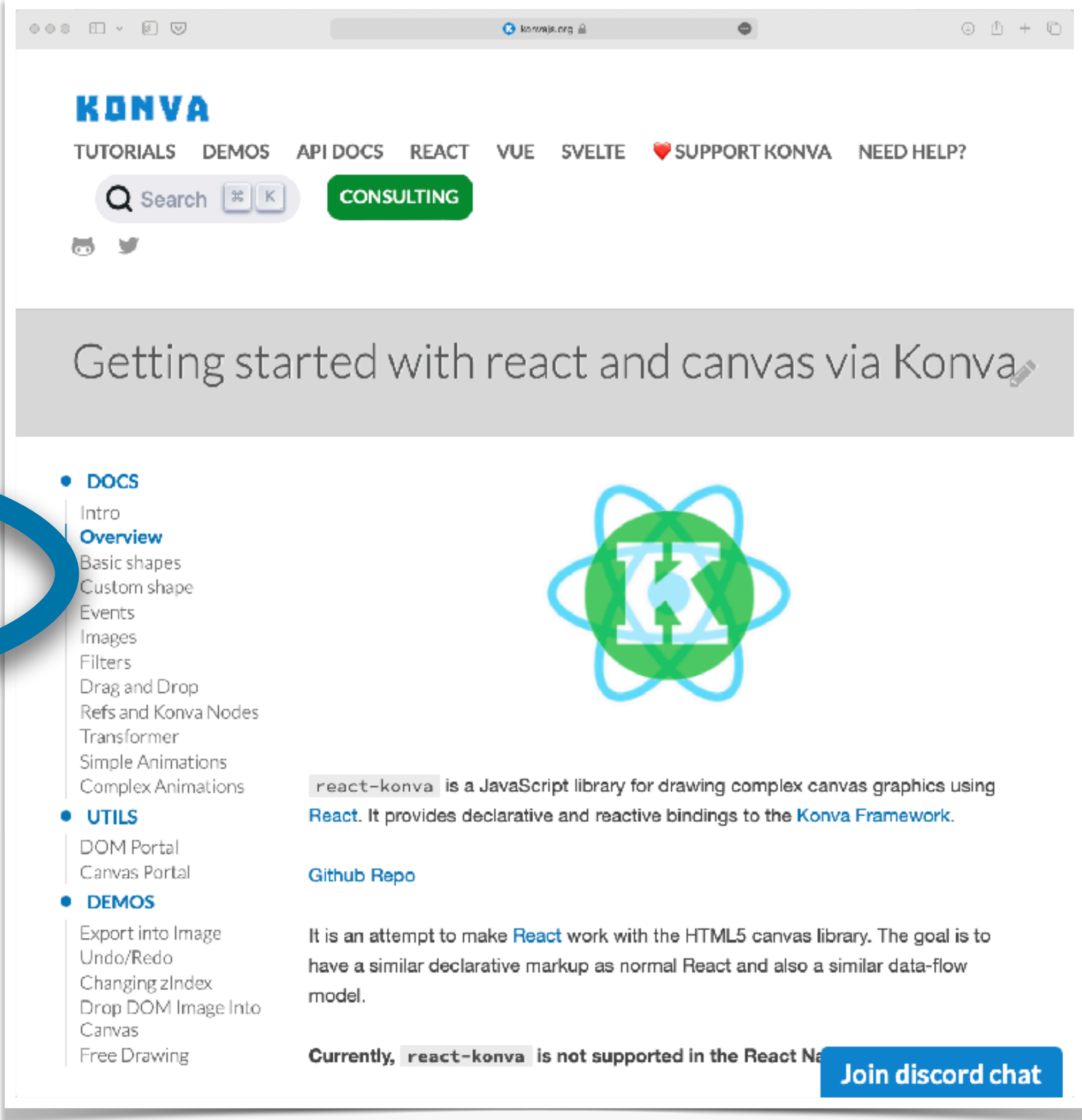
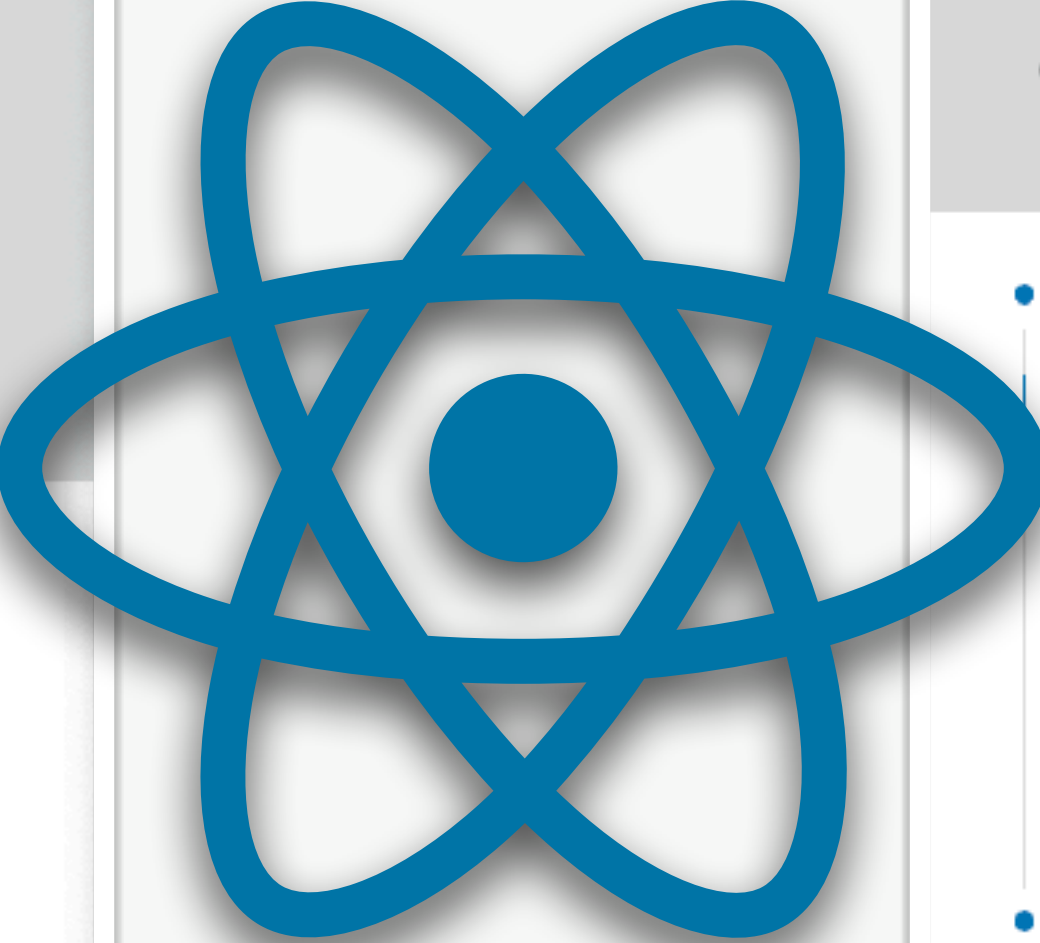
npm install konva

**GET STARTED**

### Features

- Object Oriented API with support of many shapes
- Support for desktop and mobile devices

[Join discord chat](#)



**KONVA**

TUTORIALS DEMOS API DOCS REACT VUE SVELTE ♥ SUPPORT KONVA NEED HELP?

Search  **CONSULTING**

Getting started with react and canvas via Konva

- DOCS**
  - Intro
  - Overview**
  - Basic shapes
  - Custom shape
  - Events
  - Images
  - Filters
  - Drag and Drop
  - Refs and Konva Nodes
  - Transformer
  - Simple Animations
  - Complex Animations
- UTILS**
  - DOM Portal
  - Canvas Portal
- DEMOS**
  - Export into Image
  - Undo/Redo
  - Changing zIndex
  - Drop DOM Image Into Canvas
  - Free Drawing

`react-konva` is a JavaScript library for drawing complex canvas graphics using [React](#). It provides declarative and reactive bindings to the [Konva Framework](#).

[Github Repo](#)

It is an attempt to make [React](#) work with the HTML5 canvas library. The goal is to have a similar declarative markup as normal React and also a similar data-flow model.

Currently, `react-konva` is not supported in the React Native.

[Join discord chat](#)

# Konva

```
return (  
  <Stage  
    width={gameConfig.stageDimensions.width}  
    height={gameConfig.stageDimensions.height}  
    style={{ background: 'blue', border: '1px solid black' }}  
  >  
    { /* Background graphics layer - static (no events) */ }  
    <BackgroundLayer />  
    {children}  
    { /* Dynamic App UI Layer */ }  
    <Layer>  
      <MuteButton />  
      {showGameIntro && (  
        // Play button  
        <TransparentButton  
          x={0.5 * gameConfig.stageDimensions.width - 500}  
          y={0.5 * gameConfig.stageDimensions.height - 250}  
          width={1000}  
          height={500}  
          onClick={() => appActorRef.send({ type: 'Play' })}  
        />  
      )}  
    </Layer>  
  </Stage>  
)  
);
```



# Konva

```
    { /* Background kitchen image */  
    <Group>  
      <Image  
        image={kitchenBgImage} ...  
        height={gameConfig.stageDimensions.height}  
      />  
      { /* Translucent overlay to mute the background image */  
      <Rect  
        width={gameConfig.stageDimensions.width} ...  
        y={0}  
      />  
      { /* Hen beam image */  
      <Image  
        image={henBeamImage} ...  
        y={gameConfig.henBeam.y}  
      />  
    </Group>
```

# Game Mechanics



- Who are the characters?
- How will they move?
- How will they interact?
- How will the player interact with the game?

# First Prototypes



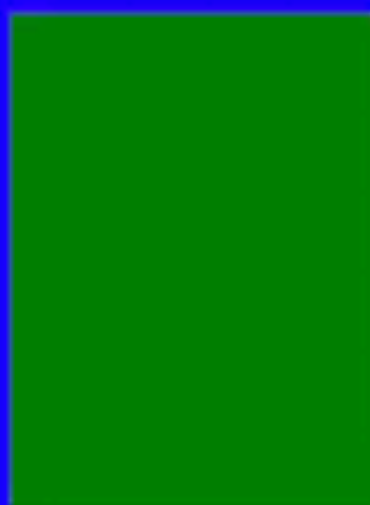


# Konva Tween

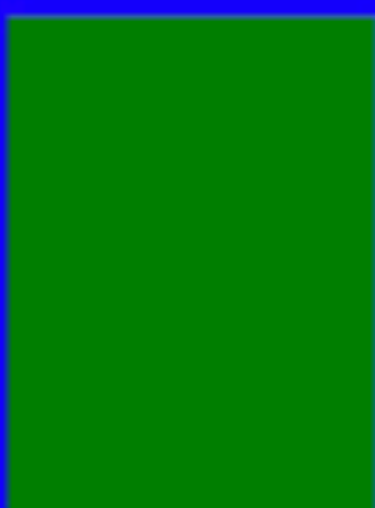
```
return new Konva.Tween({
  node: context.eggRef.current,
  duration: context.gameConfig.egg.fallingDuration,
  x: context.targetPosition.x,
  y: context.targetPosition.y,
  rotation: Math.random() > 0.5 ? 720 : -720,
  onUpdate: () => {
    if (self.getSnapshot().status === 'active') {
      self.send({
        type: 'Notify of animation position',
        position: {
          x: context.eggRef.current!.x(),
          y: context.eggRef.current!.y(),
        },
      });
    }
  },
});
```

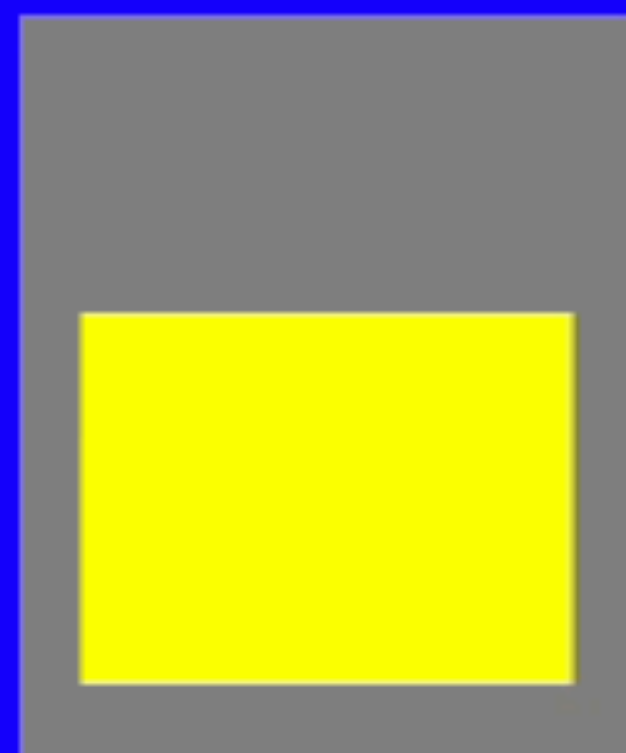






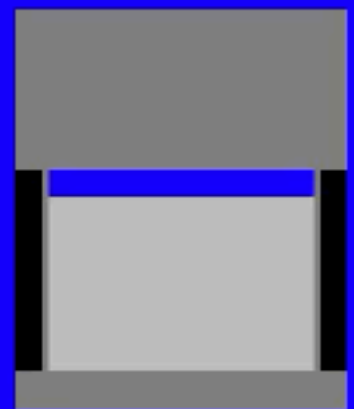


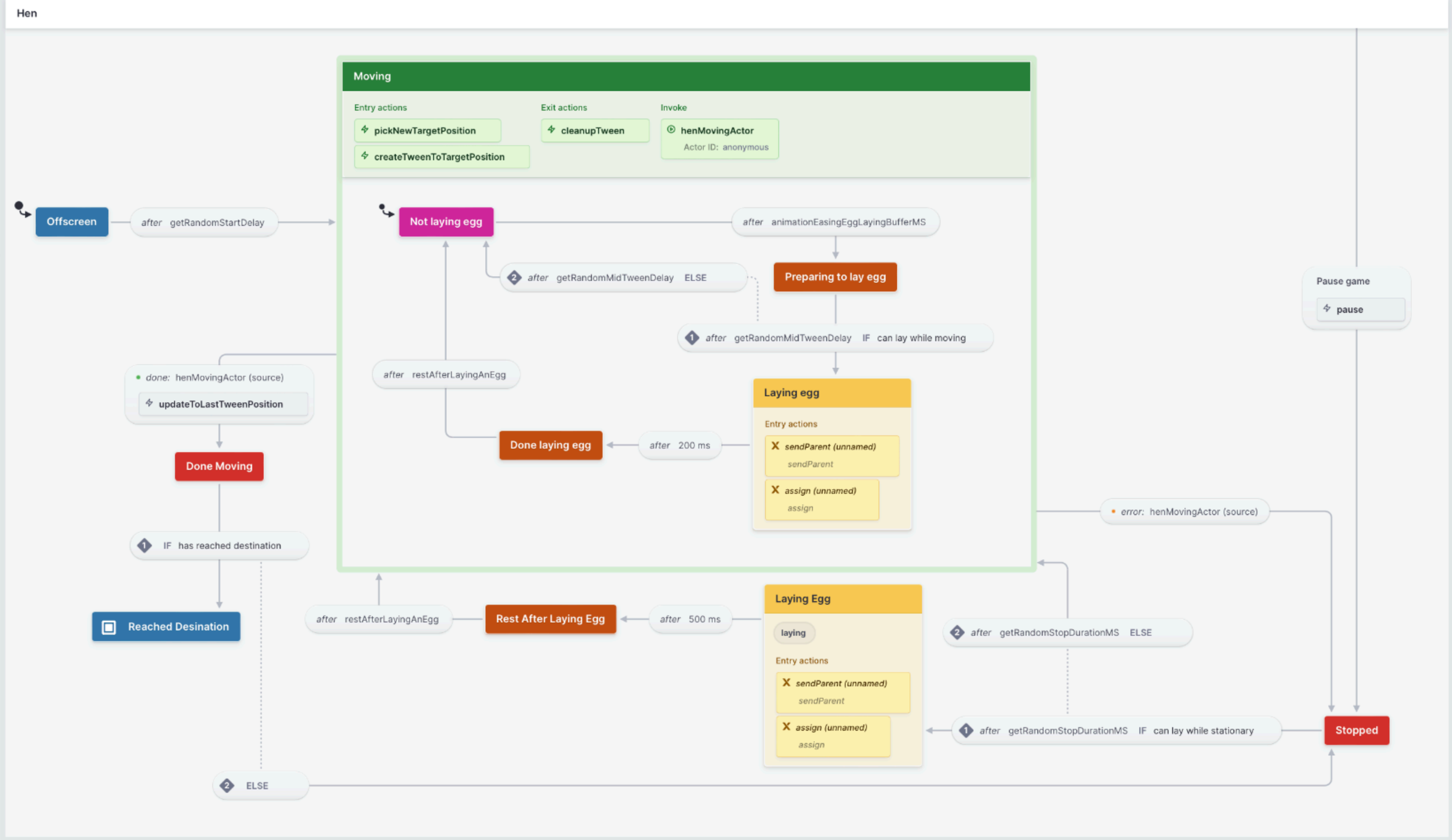




# Konva Hit Detection

- **Pixel-level** - User clicks, can include color detection
- **Bounding Box** - Fast, less-precise
- **Shape-level** - Basic shapes like rectangles, circles, polygons
- **Custom Hit Detection** - Irregular or dynamic shapes
- **Group** - Detects hits on any grouped objects `<Group>...</Group>`





My Projects

Egg Drop Public

Machines

New machine

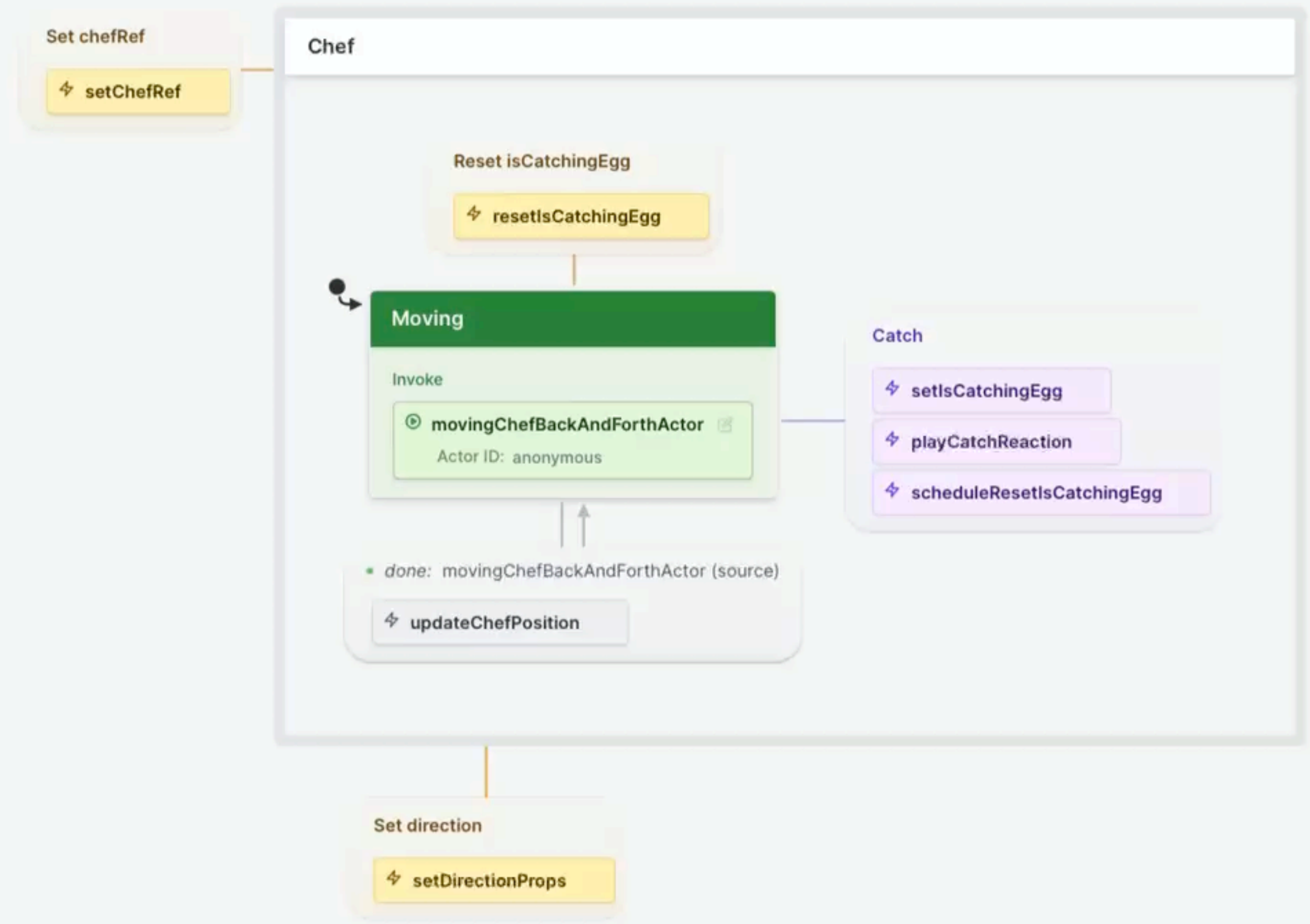
Chef

Egg

Egg Drop Game

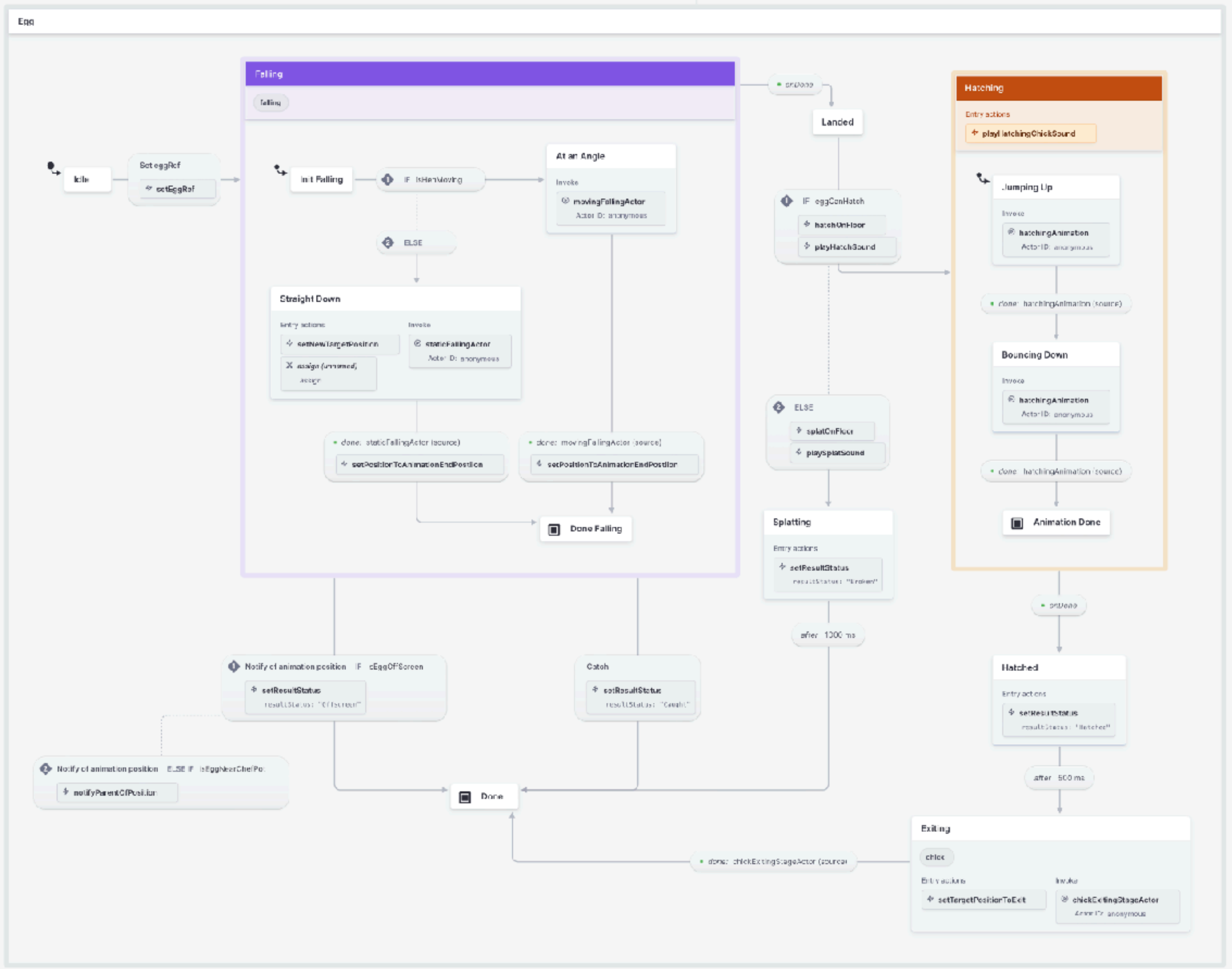
GameLevel

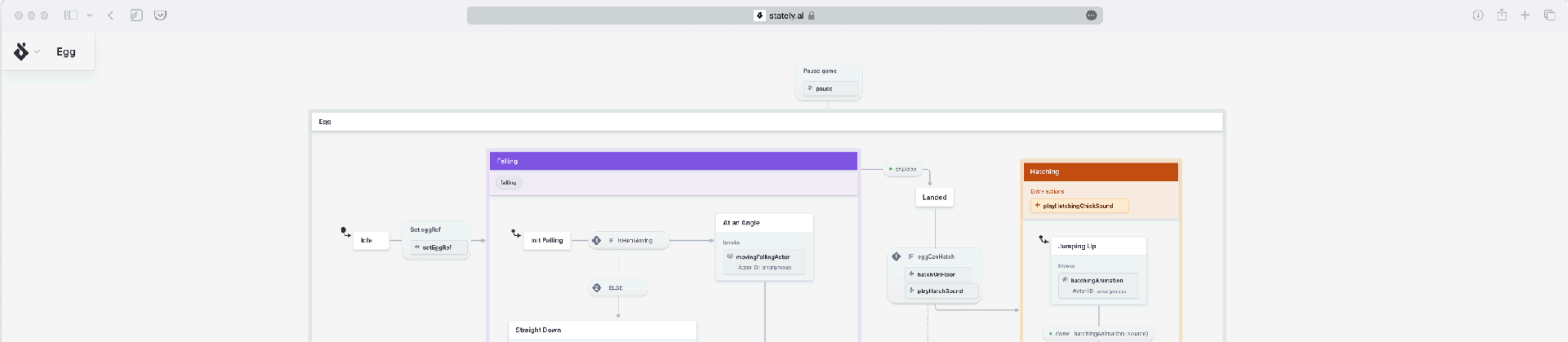
Hen



Navigation icons: Home, Search, Hand, Add

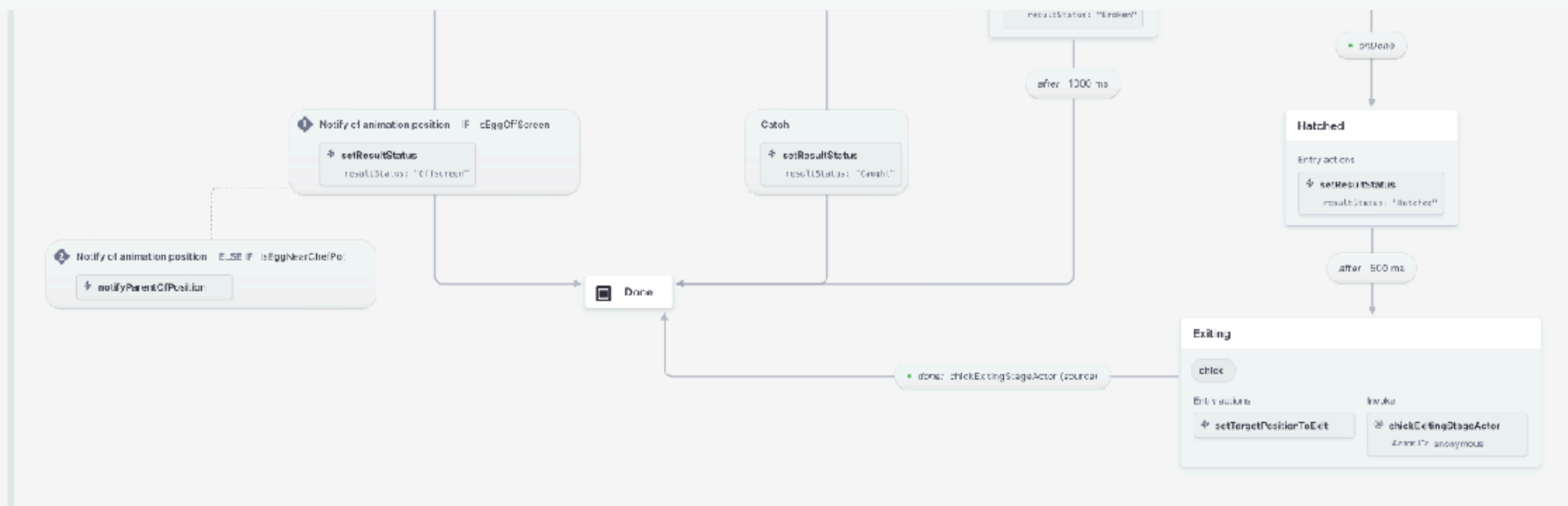
# XSTATE





# XSTATE

```
const eggState = useSelector(eggActorRef, (state) => state);
```

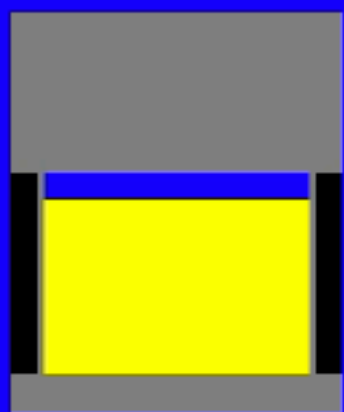






Kitchen bg

Gen: 1 Time: 0 seconds



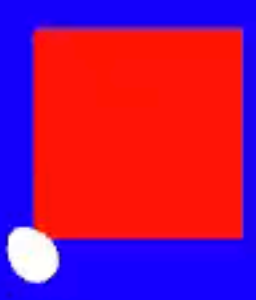


Kitchen bg



Gen: 1 Time: 25 seconds





Kitchen bg

Gen: 0 Time: 2 seconds



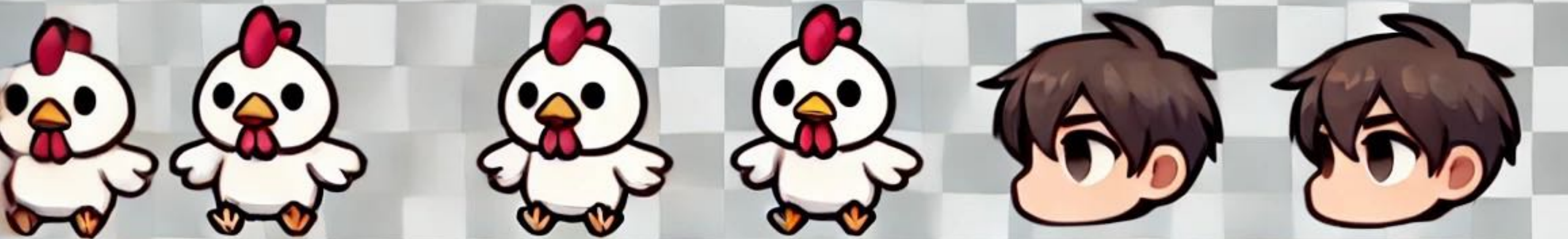
# Vector Graphics



# Vector Graphics?



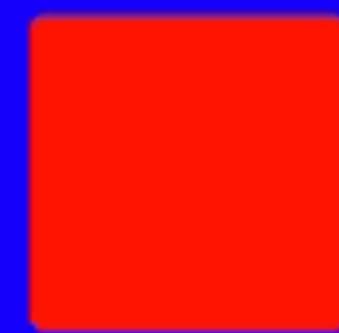
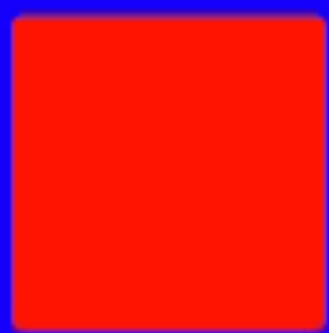
**OpenAI**











Kitchen

Gen: 1 Time: 53 seconds

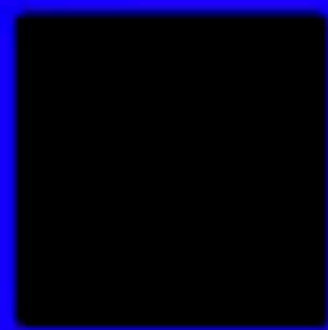


# Konva Animation

```
const animation = new Konva.Animation((frame) => {
  if (frame) {
    // Calculate new x and y positions
    const newXPos = input.node.x() + input.xSpeed;
    input.node.x(newXPos);

    // Calculate new y position with a minimum change threshold
    const minYChange = 2.5; // Minimum change in Y position to prevent it from stalling
    const deltaY = input.ySpeed * (frame.timeDiff / 1000);

    // Ensure there's always a minimum change in the Y position
    const newYPos =
      input.node.y() +
      (Math.abs(deltaY) > minYChange
        ? deltaY
        : minYChange * Math.sign(input.ySpeed));
    input.node.y(newYPos);
  }
});
```



Kitchen



Gen: 0 Time: 6 seconds



Gen: 0 Time: 17 seconds

Score: 0

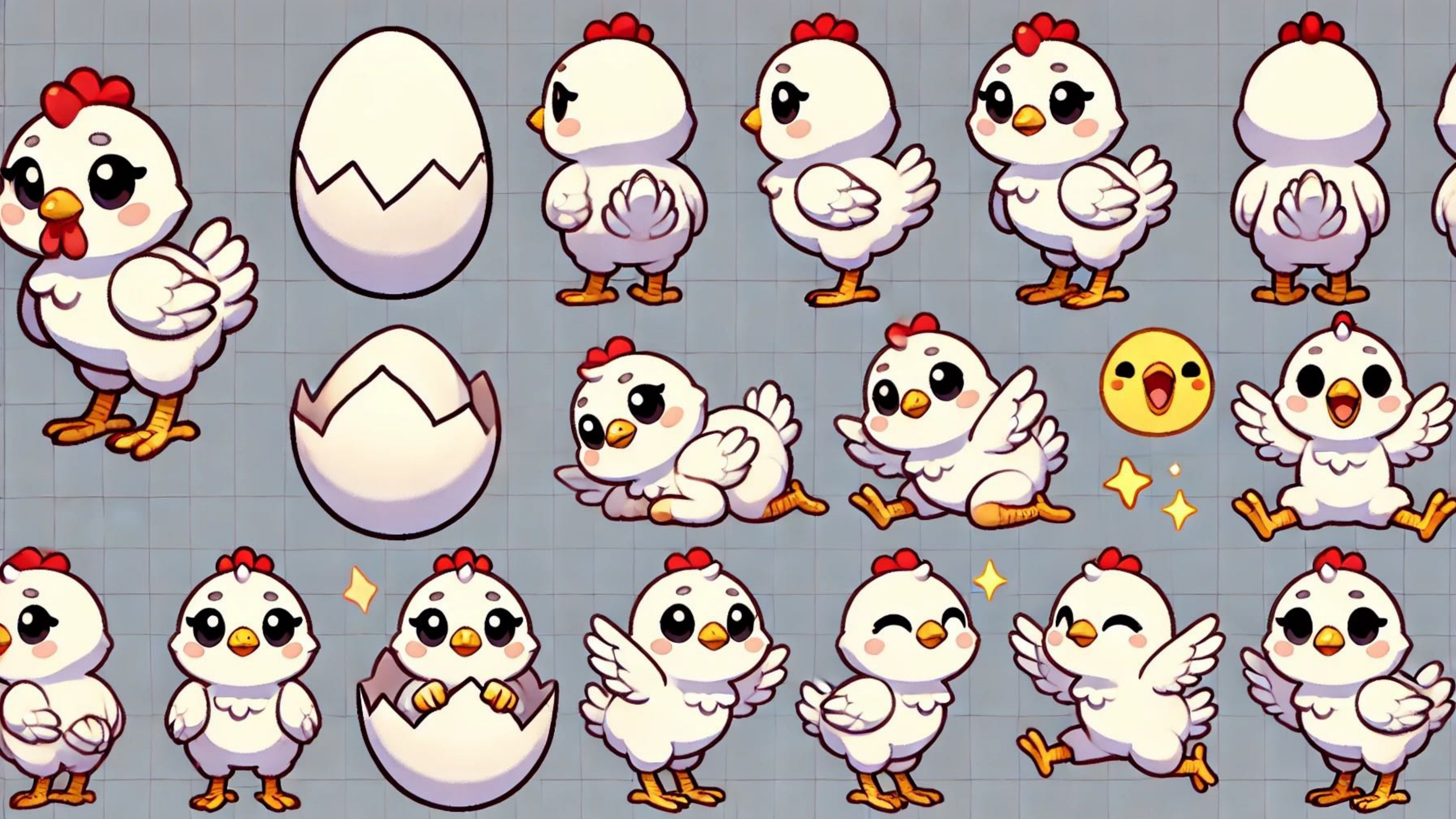
Eggs: 0

Gold: 0



# Open AI Casting Call for Hens

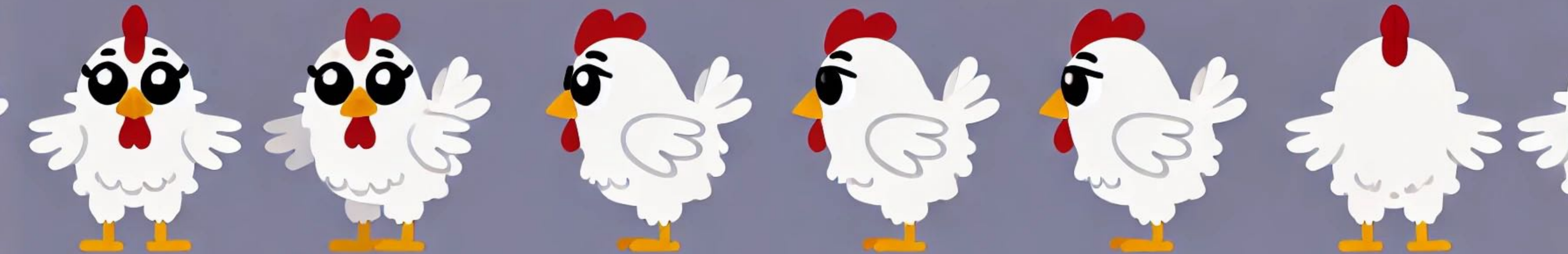














# Logoist 5

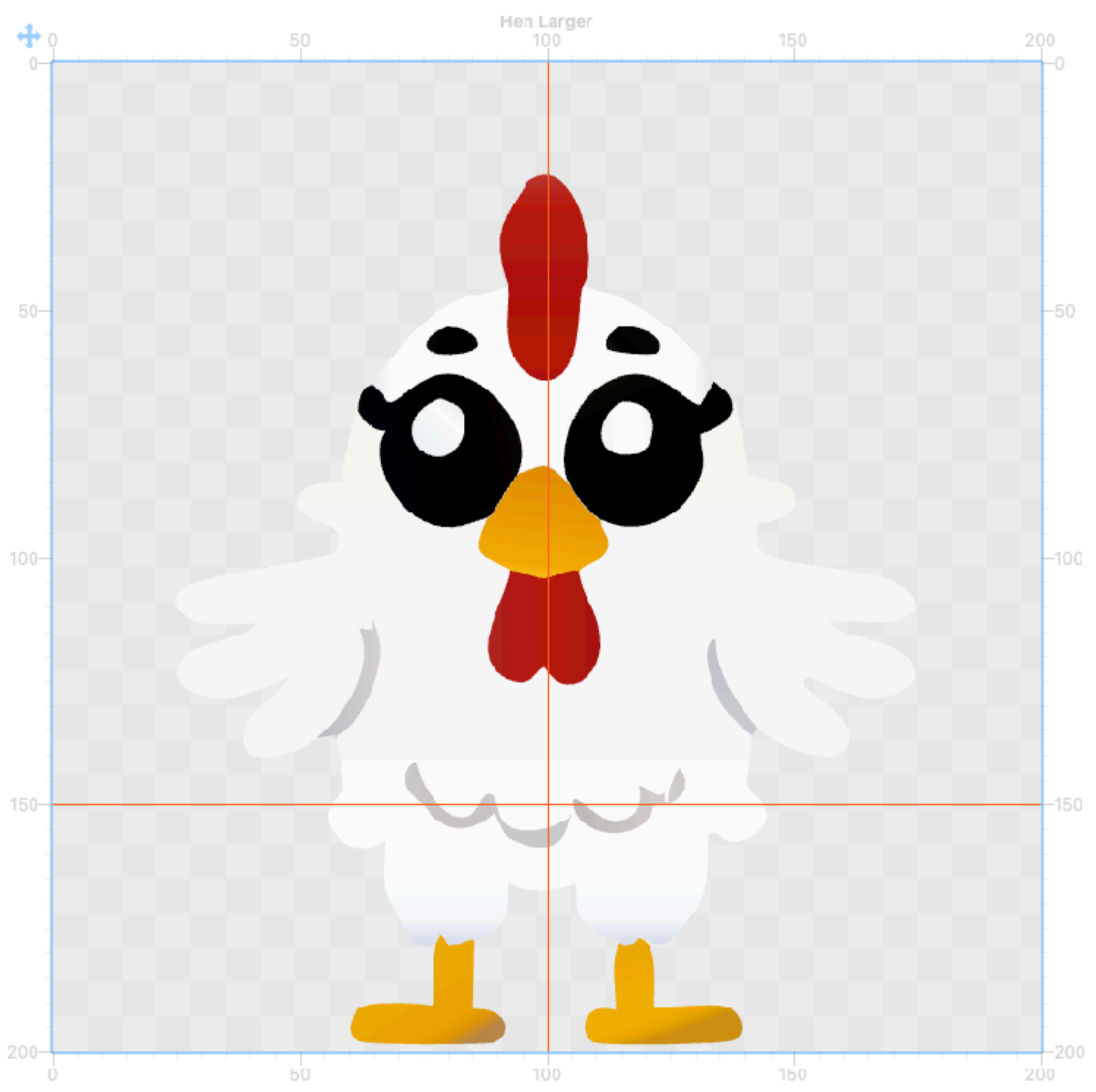


↓ **Download Demo for Mac**  
Version 5.1.2

🛒 **Buy Logoist 5**  
US\$ 39.99 on the App Store

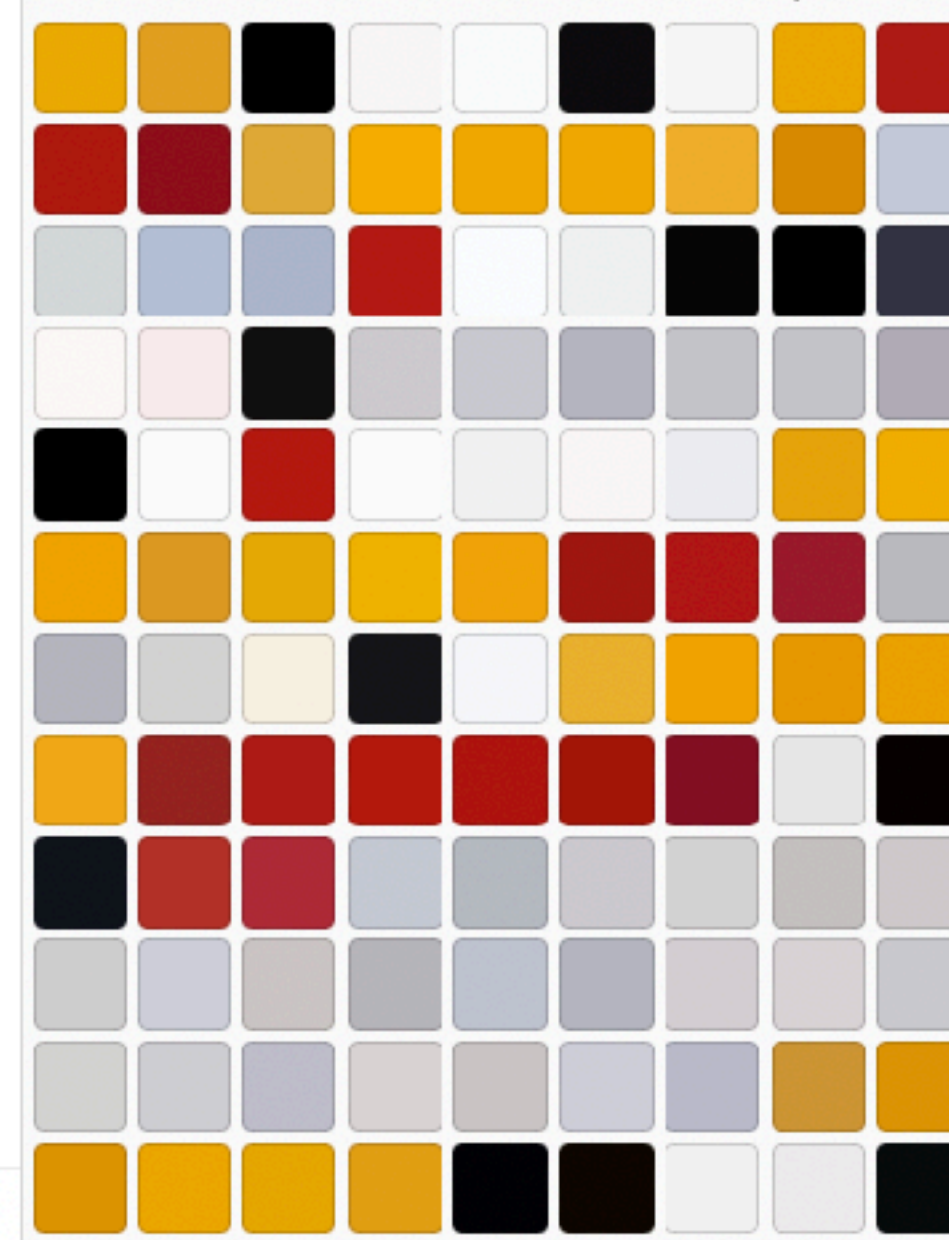
- Text
- Shape
- Line
- Path
- VectorAI
- LiveShape
- Clipart
- Lens Flare
- Templates
- Image
- Background
- Group
- Instance
- Embed

- ▶ Egg Glow
- ▶ Chick
- ▶ Egg Broken
- ▶ Title Page
- ▼ Hen Larger
  - ▶ forward (17 Objects)
  - ▶ angle-left (15 Objects)
  - ▶ angle-right (15 Objects)
  - ▶ walk-left-1 (12 Objects)
  - ▶ walk-left-2 (12 Objects)
  - ▶ walk-left-3 (12 Objects)
  - ▶ walk-left-4 (12 Objects)
  - ▶ walk-right-1 (12 Objects)
  - ▶ walk-right-2 (12 Objects)
  - ▶ walk-right-3 (12 Objects)
  - ▶ walk-right-4 (12 Objects)
  - ▶ back-left (8 Objects)
  - ▶ back-right (8 Objects)
  - ▶ jump-1 (14 Objects)
  - ▶ jump-2 (12 Objects)
- ▶ Logo Large
- ▶ Logo Square
- ▶ Favicon
- ▶ Points



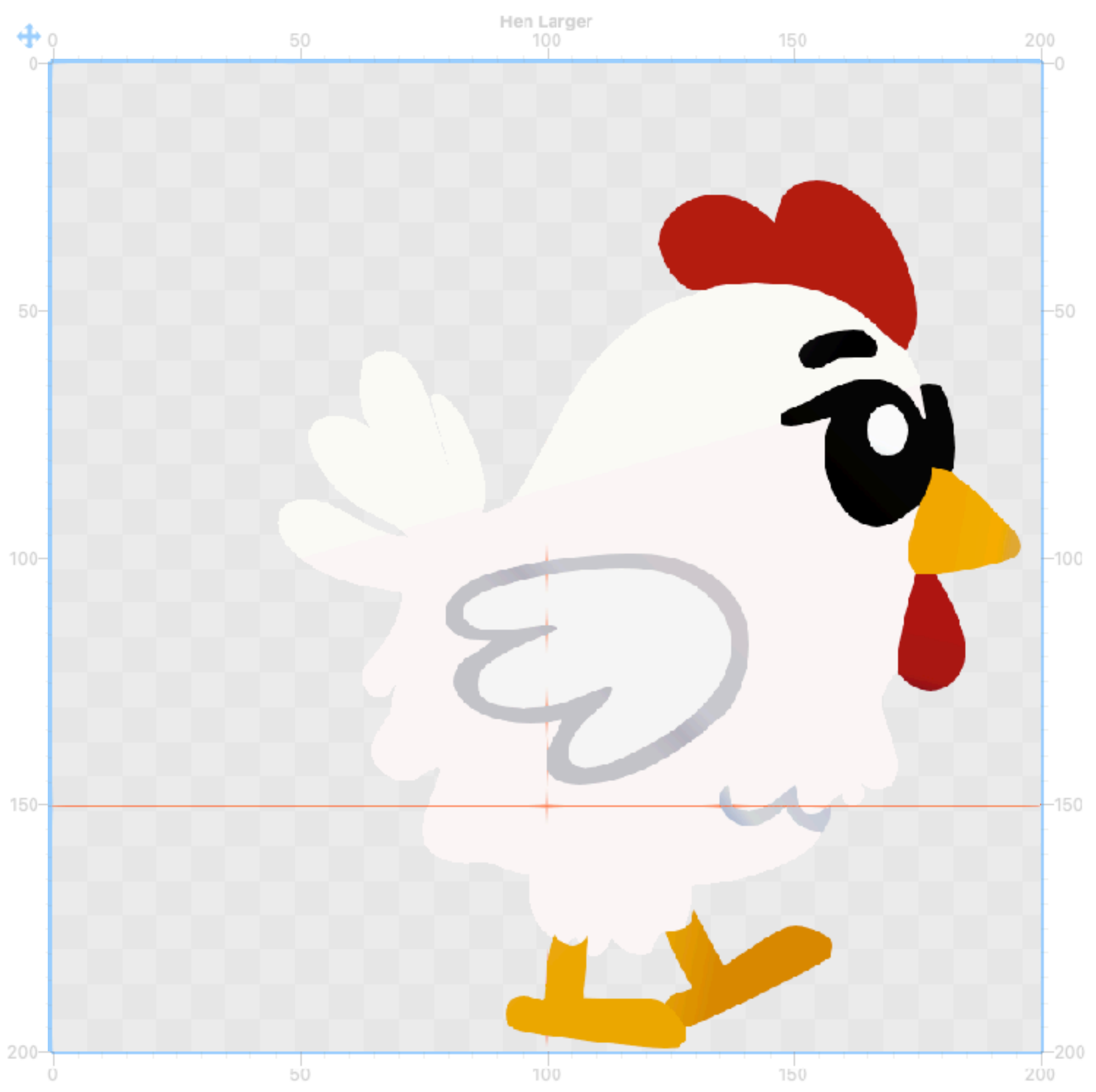
Please select objects on the artboard to change properties, or select style presets

Colors on this Artboard: Sort by References



- Text
- Shape
- Line
- Path
- VectorAI
- LiveShape
- Clipart
- Lens Flare
- Templates
- Image
- Background
- Group
- Instance
- Embed

- ▶ Egg Glow
- ▶ Chick
- ▶ Egg Broken
- ▶ Title Page
- ▼ Hen Larger
  - ▶ forward (17 Objects)
  - ▶ angle-left (15 Objects)
  - ▶ angle-right (15 Objects)
  - ▶ walk-left-1 (12 Objects)
  - ▶ walk-left-2 (12 Objects)
  - ▶ walk-left-3 (12 Objects)
  - ▶ walk-left-4 (12 Objects)
  - ▶ walk-right-1 (12 Objects)
  - ▶ walk-right-2 (12 Objects)
  - ▶ walk-right-3 (12 Objects)
  - ▶ walk-right-4 (12 Objects)
  - ▶ back-left (8 Objects)
  - ▶ back-right (8 Objects)
  - ▶ jump-1 (14 Objects)
  - ▶ jump-2 (12 Objects)
- ▶ Logo Large
- ▶ Logo Square
- ▶ Favicon
- ▶ Points



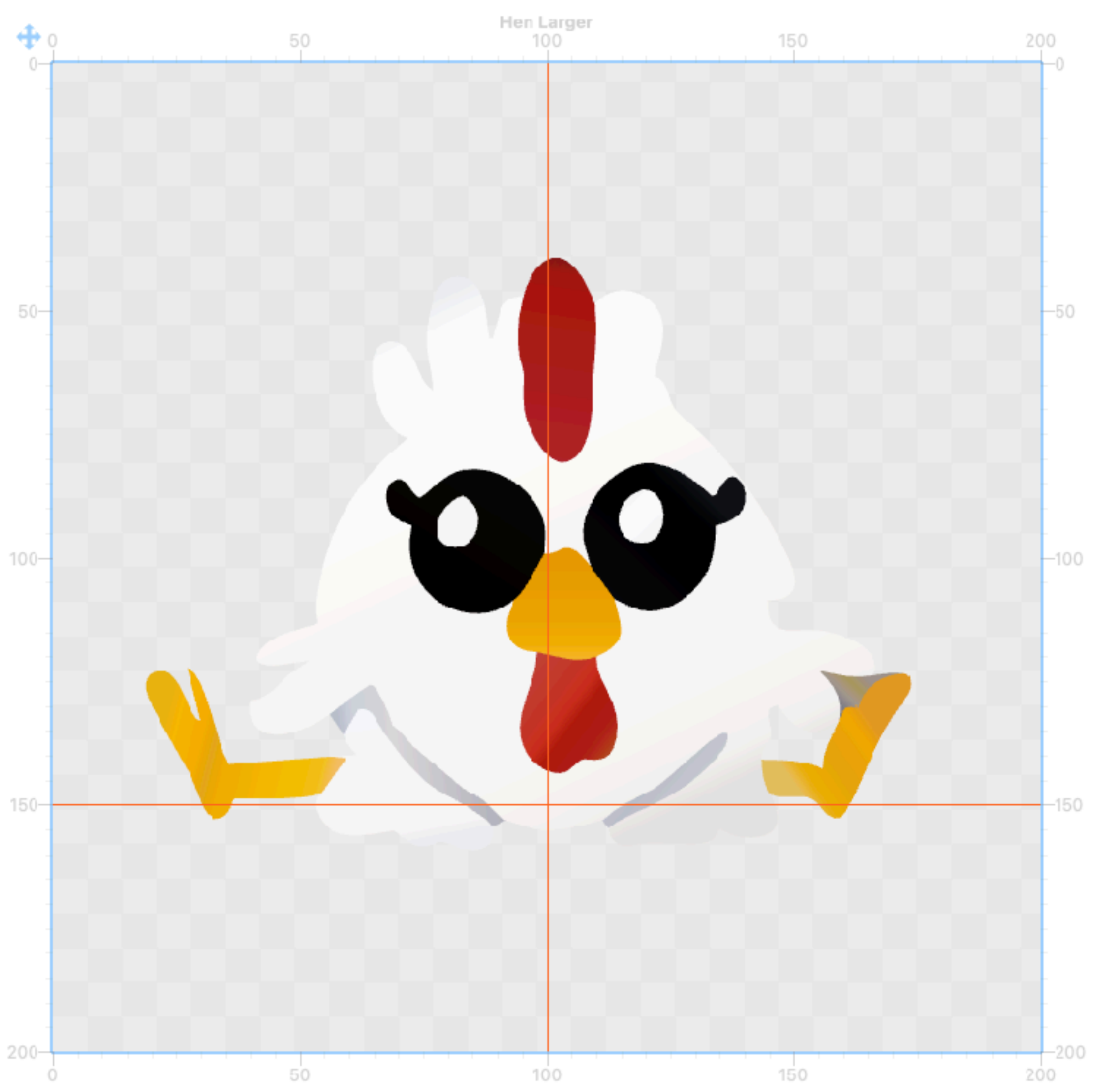
Please select objects on the artboard to change properties, or select style presets

Colors on this Artboard: Sort by References



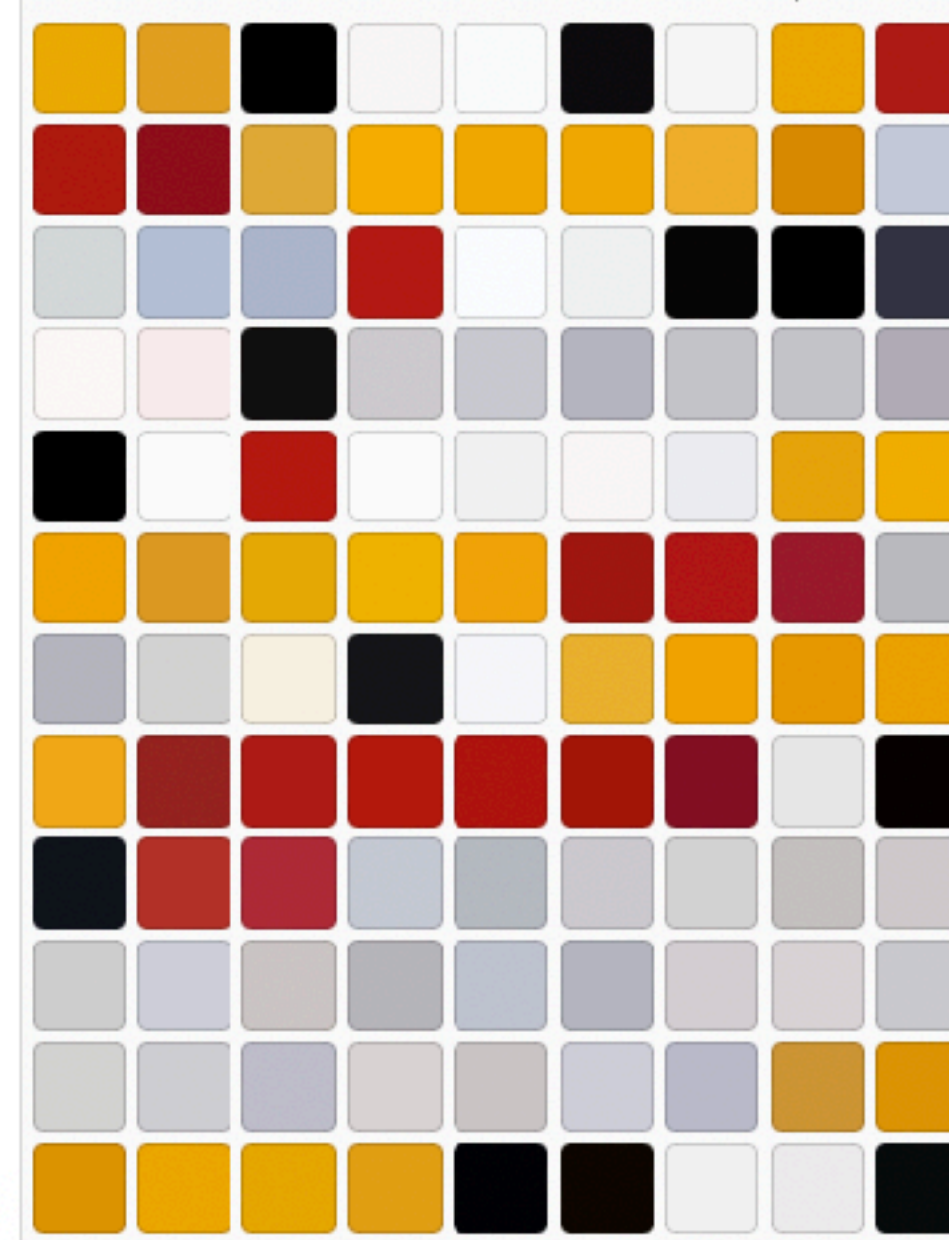
- Text
- Shape
- Line
- Path
- VectorAI
- LiveShape
- Clipart
- Lens Flare
- Templates
- Image
- Background
- Group
- Instance
- Embed

- ▶ Egg Glow
- ▶ Chick
- ▶ Egg Broken
- ▶ Title Page
- ▼ Hen Larger
  - ▶ forward \*7 Objects
  - ▶ angle-left \*5 Objects
  - ▶ angle-right \*6 Objects
  - ▶ walk-left-1 \*2 Objects
  - ▶ walk-left-2 \*2 Objects
  - ▶ walk-left-3 \*2 Objects
  - ▶ walk-left-4 \*2 Objects
  - ▶ walk-right-1 \*2 Objects
  - ▶ walk-right-2 \*2 Objects
  - ▶ walk-right-3 \*2 Objects
  - ▶ walk-right-4 \*2 Objects
  - ▶ back-left 6 Objects
  - ▶ back-right 6 Objects
  - ▶ jump-1 \*4 Objects
  - ▶ jump-2 \*2 Objects
- ▶ Logo Large
- ▶ Logo Square
- ▶ Favicon
- ▶ Points



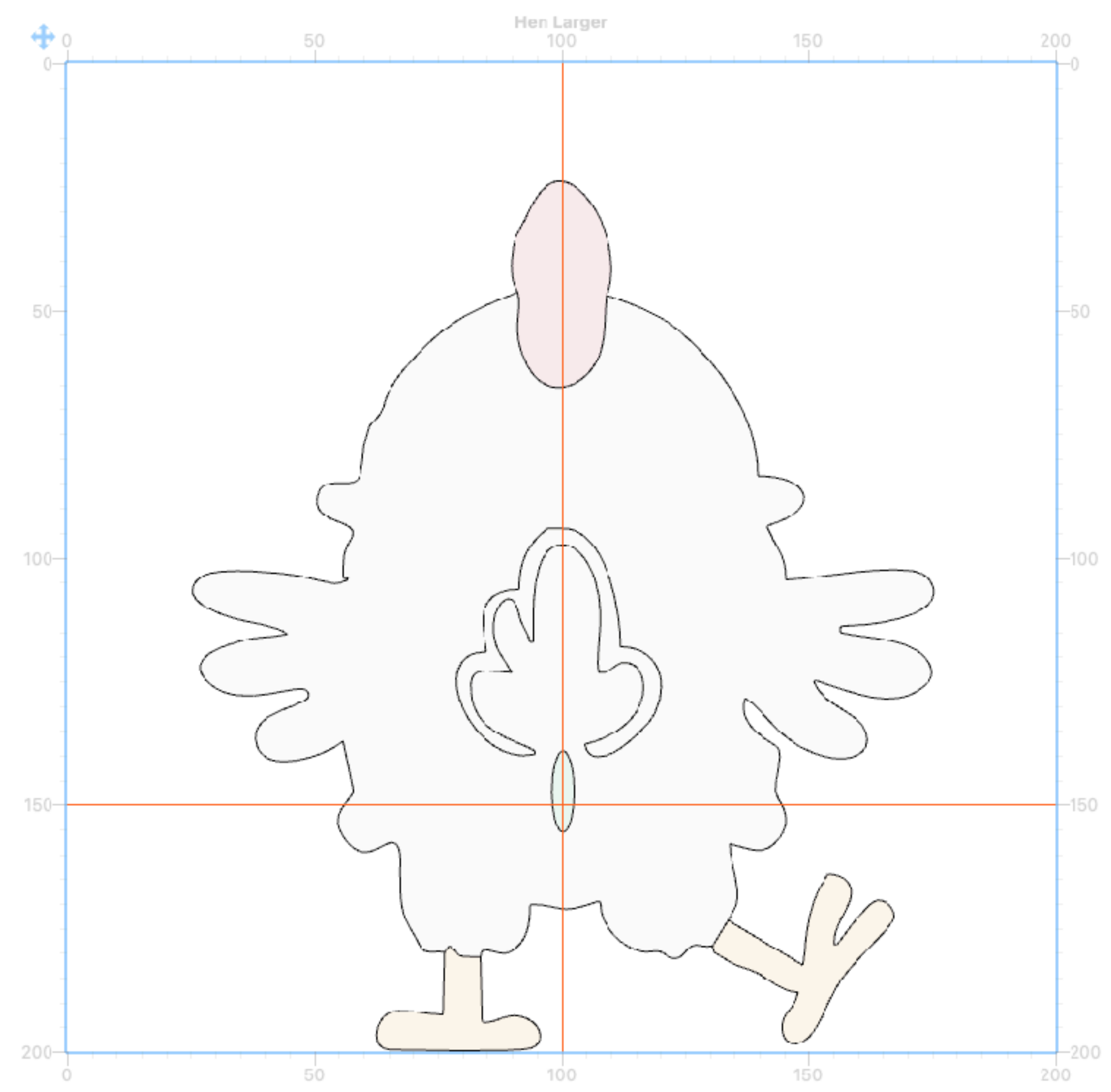
Please select objects on the artboard to change properties, or select style presets

Colors on this Artboard: Sort by References



- Text
- Shape
- Line
- Path
- VectorAI
- LiveShape
- Clipart
- Lens Flare
- Templates
- Image
- Background
- Group
- Instance
- Embed

- ▶ Egg Glow
- ▶ Chick
- ▶ Egg Broken
- ▶ Title Page
- ▼ Hen Larger
  - ▶ forward \*7 Objects
  - ▶ angle-left \*5 Objects
  - ▶ angle-right \*6 Objects
  - ▶ walk-left-1 \*2 Objects
  - ▶ walk-left-2 \*2 Objects
  - ▶ walk-left-3 \*2 Objects
  - ▶ walk-left-4 \*2 Objects
  - ▶ walk-right-1 \*2 Objects
  - ▶ walk-right-2 \*2 Objects
  - ▶ walk-right-3 \*2 Objects
  - ▶ walk-right-4 \*2 Objects
  - ▶ back-left 6 Objects
  - ▶ back-right 6 Objects
  - ▶ jump-1 \*4 Objects
  - ▶ jump-2 \*2 Objects
- ▶ Logo Large
- ▶ Logo Square
- ▶ Favicon
- ▶ Points



Please select objects on the artboard to change properties, or select style presets

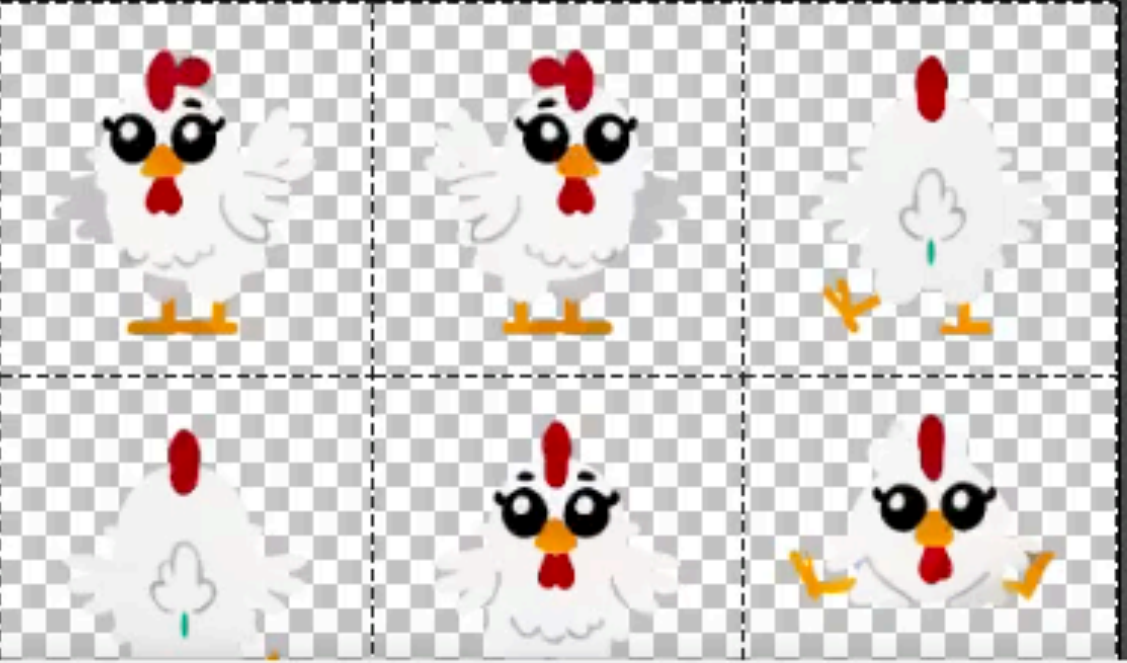
Colors on this Artboard: Sort by References



# Texture Packer



- ### Sprites
- angle-left.png
  - angle-right.png
  - back-left.png
  - back-right.png
  - forward.png
  - jump-1.png
  - jump-2.png
  - walk-left-1.png
  - walk-left-2.png
  - walk-left-3.png
  - walk-left-4.png
  - walk-right-1.png
  - walk-right-2.png
  - walk-right-3.png
  - walk-right-4.png



Animation preview

Speed  7 FPS

### Settings

Output files

Framework: JSON (Hash)

Data file: ation/Hen, Egg, Chick/hen.sprite.

Texture file: hen.sprite.png

---

Image format

Texture format: PNG-32

Pixel format: RGBA8888

Scaling variants:

---

Packing

Algorithm: MaxRects

Max size: 2048

Trim mode: None

Multi pack: Off

Extrude: 1



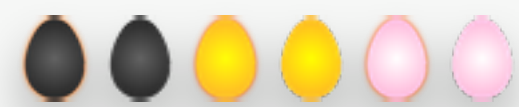


Animation preview

A large yellow chick running on a black background, shown in a full frame within a preview window. The chick has a happy expression with closed eyes and is in a dynamic running pose.

Speed  5 FPS





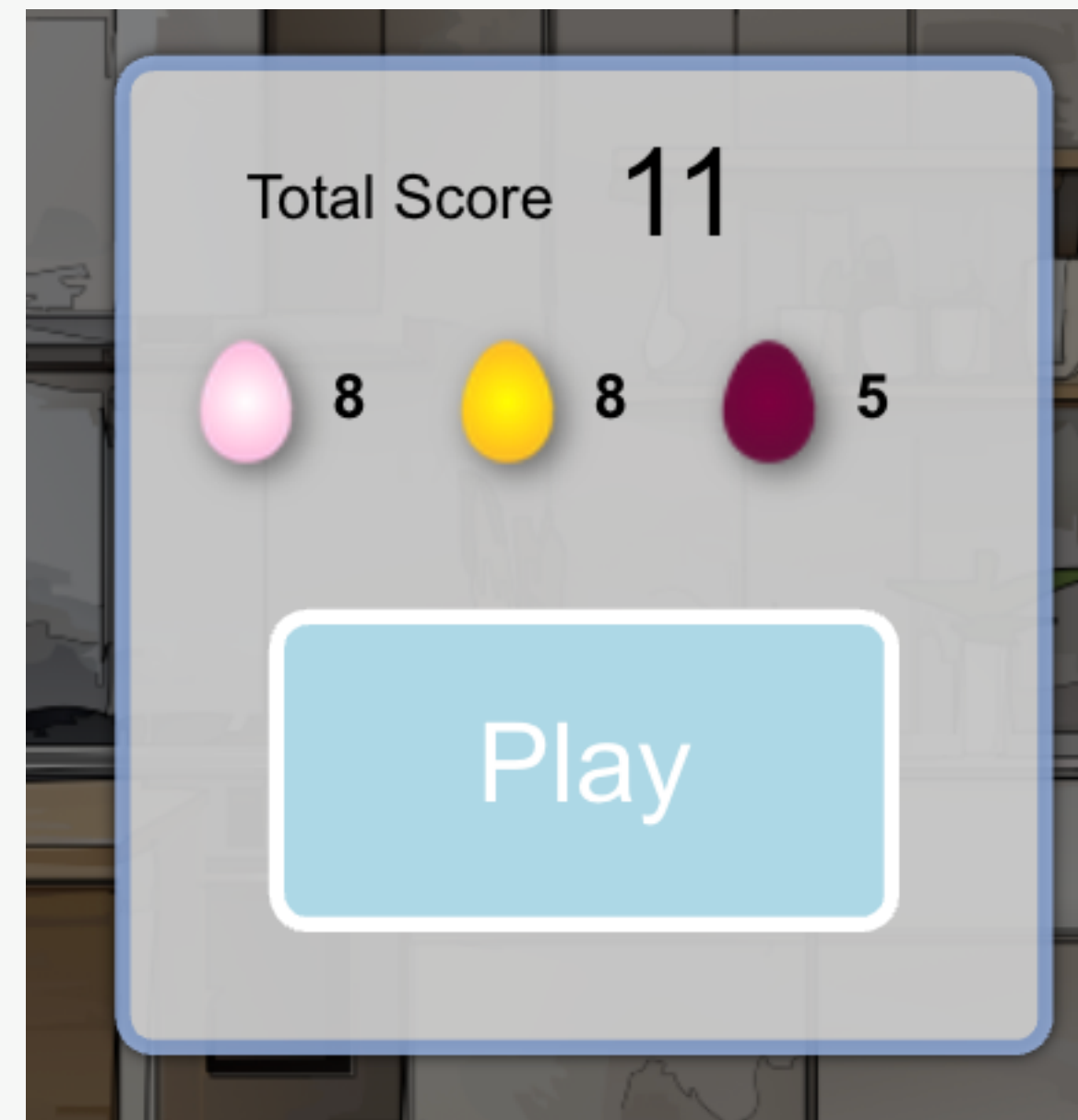
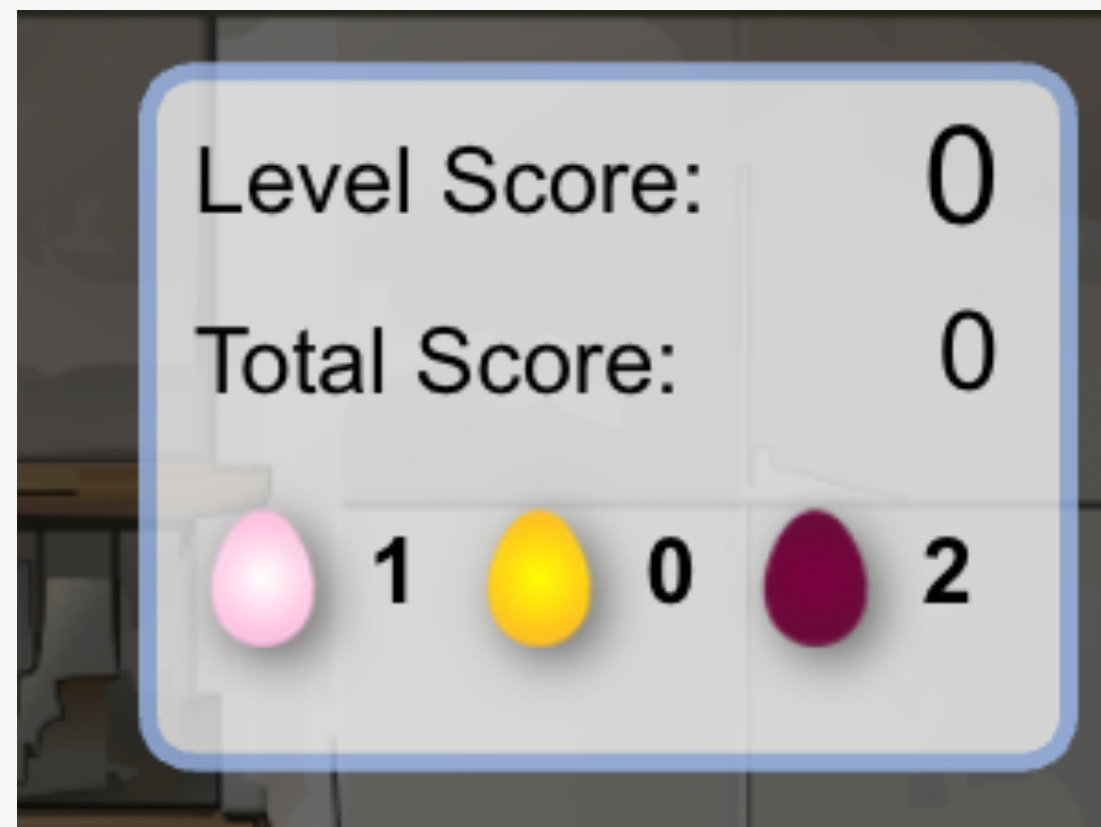






# What about fonts?

**Arial** in a game is so sad



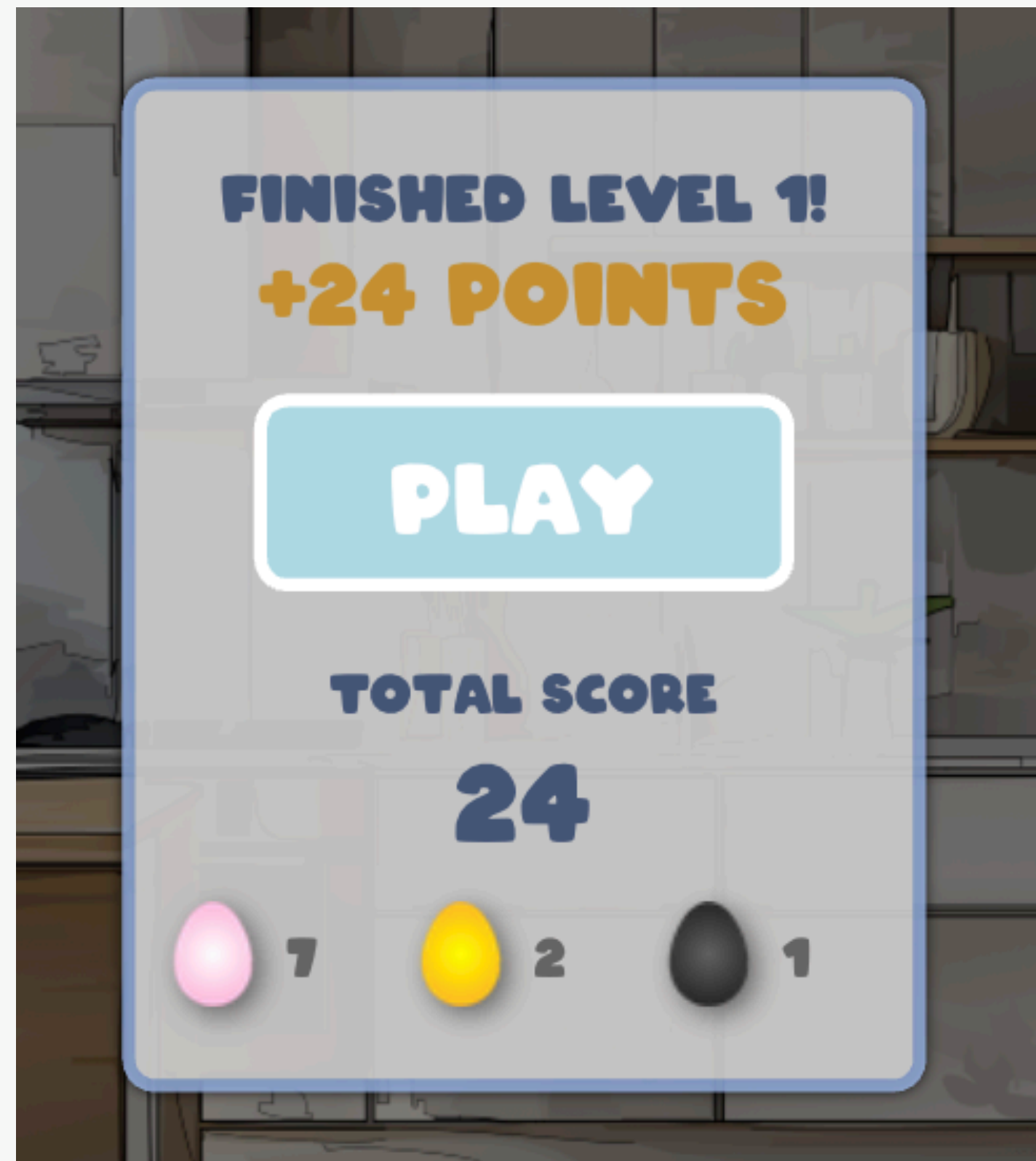
# ARCO

**THE QUICK YELLOW CHICK JUMPS OVER THE LAZY CHEF.**

```
@font-face {  
  font-family: 'Arco';  
  src: local('Arco-Regular'), url('/fonts/ARCO.ttf') format('truetype');  
}
```

# ARCO

**THE QUICK YELLOW CHICK JUMPS OVER THE LAZY CHEF.**





# ARCO



# ARCO



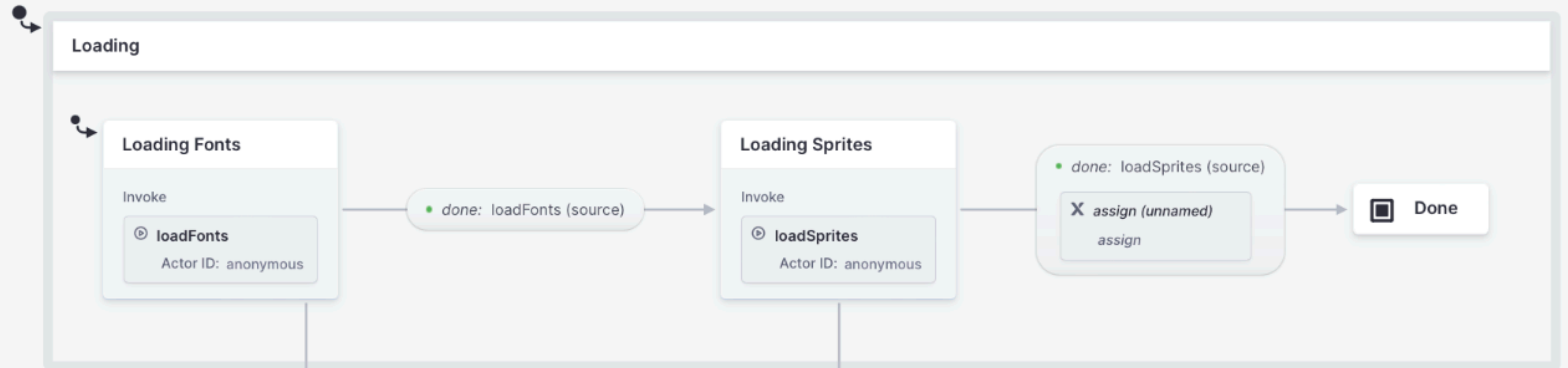
**FOUT** - *Flash of unstyled text*

**FOIT** - *Flash of invisible text*

**FOFT** - *Flash of faux text*

# ARCO

```
new FontFaceObserver('Arco');
```



```
loadFonts: fromPromise(() => {  
  const arcoFont = new FontFaceObserver('Arco');  
  const jetBrainsMonoFont = new FontFaceObserver('JetBrains Mono');  
  return Promise.all([arcoFont.load(), jetBrainsMonoFont.load()]);  
}),
```



# EGG DROP

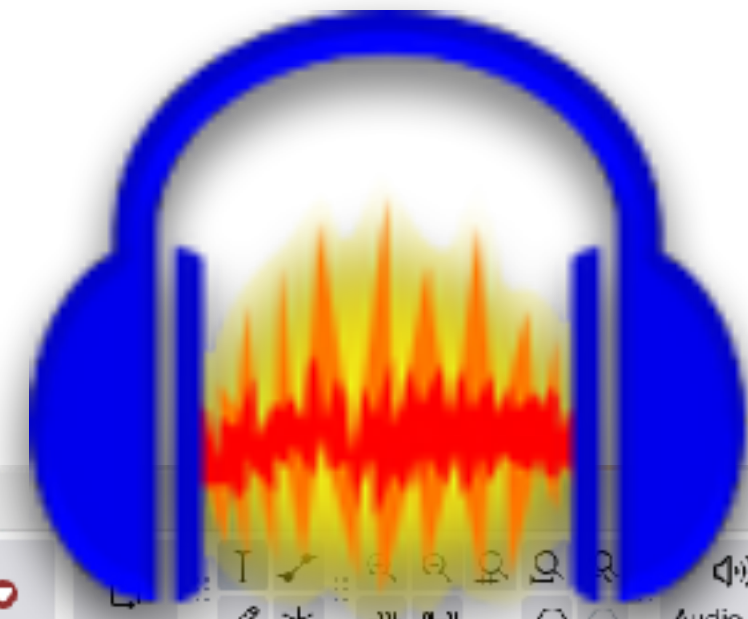
W'SUP!?



# EGG DROP

W'SUP!?





# Audacity

The screenshot shows the Audacity application window. At the top, there is a toolbar with various icons for playback (stop, play, record), editing (cut, copy, paste, undo, redo), and audio processing (normalize, fade in, fade out). Below the toolbar is a time ruler with markers every 0.05 seconds, ranging from 0.00 to 0.60. The main workspace contains a single audio track named 'oh-no1'. The track's volume is set to 1.0, and it has a mute button and an effects slot. The waveform shows a complex audio signal. At the bottom, the transport controls are visible, including a tempo of 120, a time signature of 4/4, a snap setting of 1/8, and a selection range from 00 h 00 m 00.000 s to 00 h 00 m 00.000 s. The status bar at the very bottom indicates the software is 'Stopped'.



# HOWLER.JS

```
export const sounds = {
  backgroundLoop: new Howl({
    src: ['sounds/i-am-dreaming-or-final-fantasy-menu-kinda-thing-29173.mp3'],
    volume: 0.5,
    loop: true,
  }),
  layEgg: new Howl({
    src: ['sounds/laid.wav'],
    volume: 0.4,
  }),
  catch: new Howl({
    src: ['sounds/marimba-c5.wav'],
    volume: 0.5,
  }),
  hatch: new Howl({
    src: ['sounds/egg-crack.mp3'],
    volume: 0.5,
  })
}
```



# HOWLER.JS

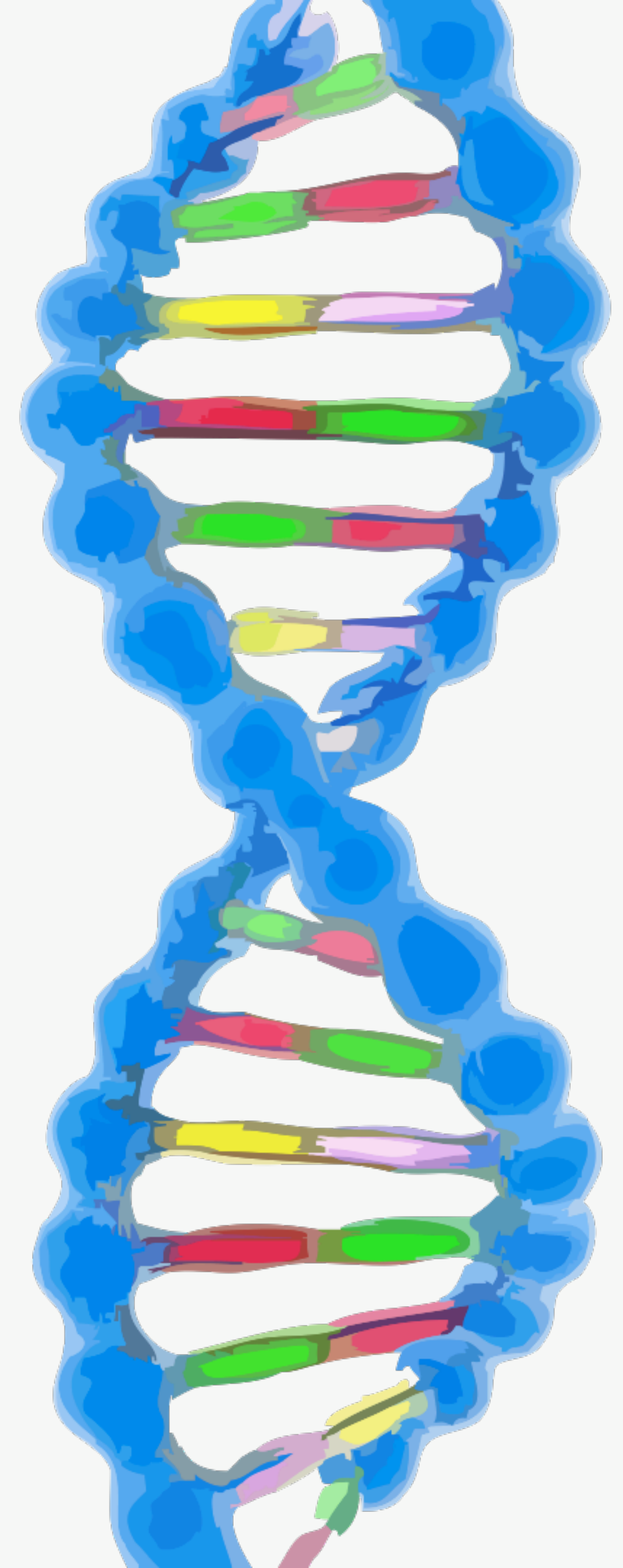
+

# XSTATE

```
// Sounds
playSplatSound: () => {
  sounds.splat.play();
},
playHatchSound: () => {
  sounds.hatch.play();
},
playHatchingChickSound: ({ context }) => {
  switch (context.color) {
    case 'gold':
      sounds.yipee.play();
      break;
    case 'white':
      sounds.haha.play();
  }
}
```



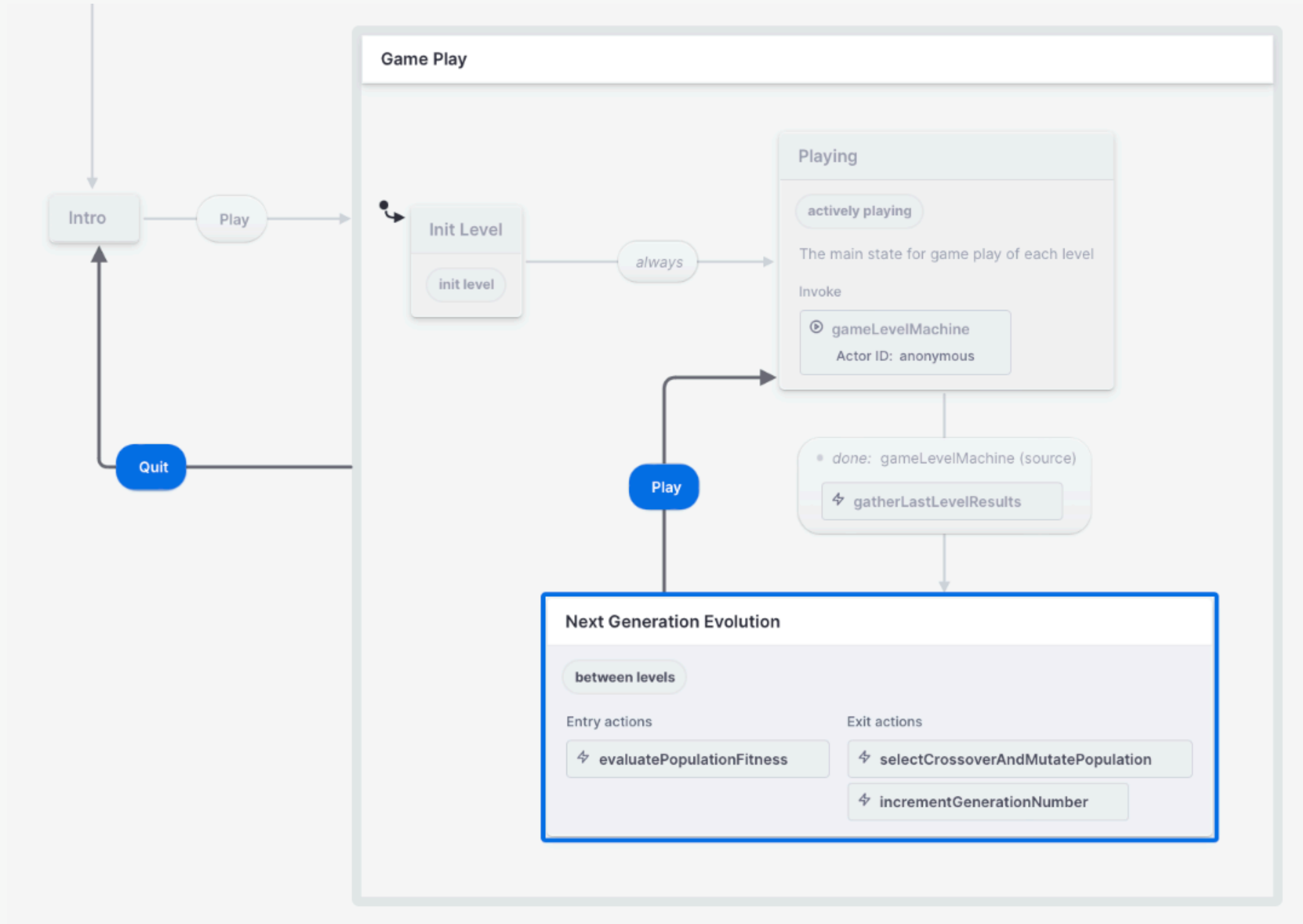
# **GENETIC ALGORITHM IN EGG DROP**



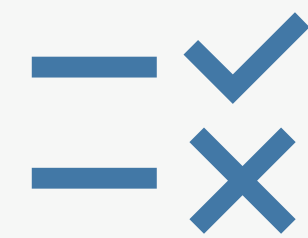
# HENINDIVIDUAL



```
export interface Hendividual {  
  id: string;  
  // Configuration  
  initialPosition: Position;  
  speed: number;  
  baseTweenDurationSeconds: number;  
  maxEggs: number;  
  stationaryEggLayingRate: number;  
  movingEggLayingRate: number;  
  restAfterLayingEggMS: number;  
  blackEggRate: number;  
  goldEggRate: number;  
  hatchRate: number;  
  minX: number;  
  maxX: number;  
  minStopMS: number;  
  maxStopMS: number;
```



# Evaluation & Fitness



Evaluate performance

Hens that lay more eggs



Reward behavior



Hens whose eggs go uncaught

Hens whose black eggs get caught



Punishment



Hens who don't lay any eggs at all



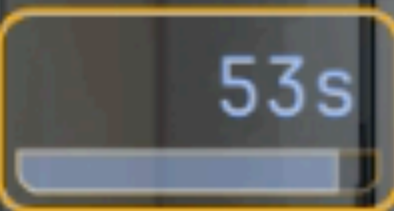
Weighted criteria



LEVEL 1

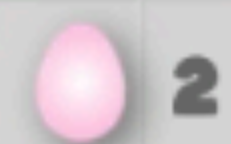


53s



SCORE

2



2



x



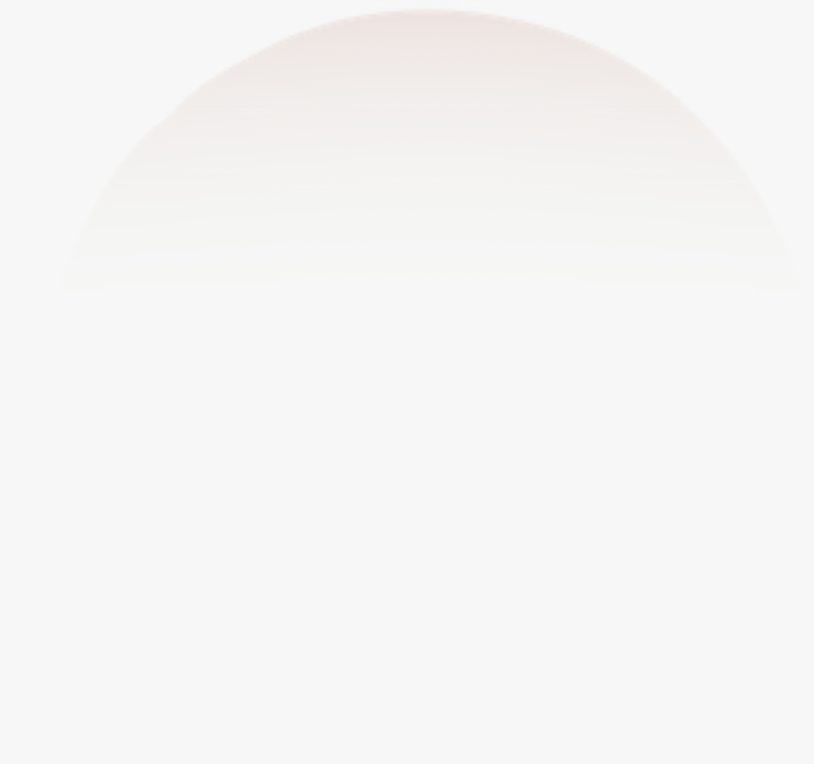
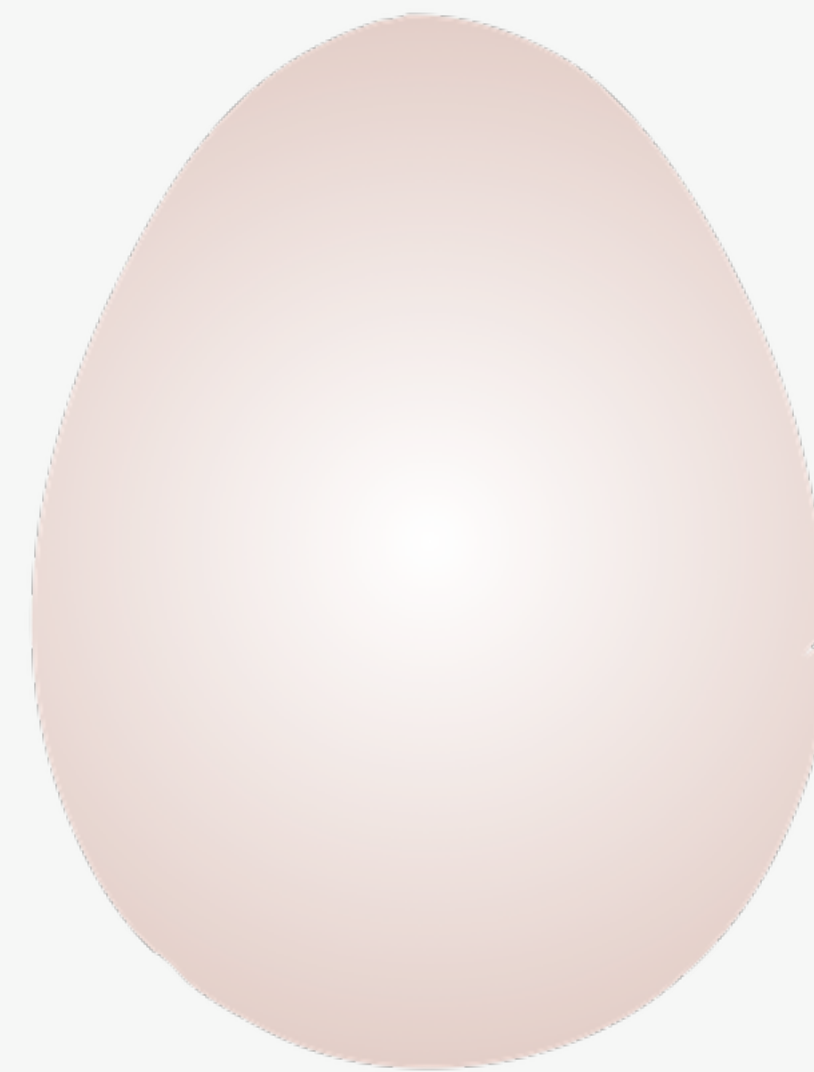
x

TOTAL SCORE

2



**TWEAK THE GA**

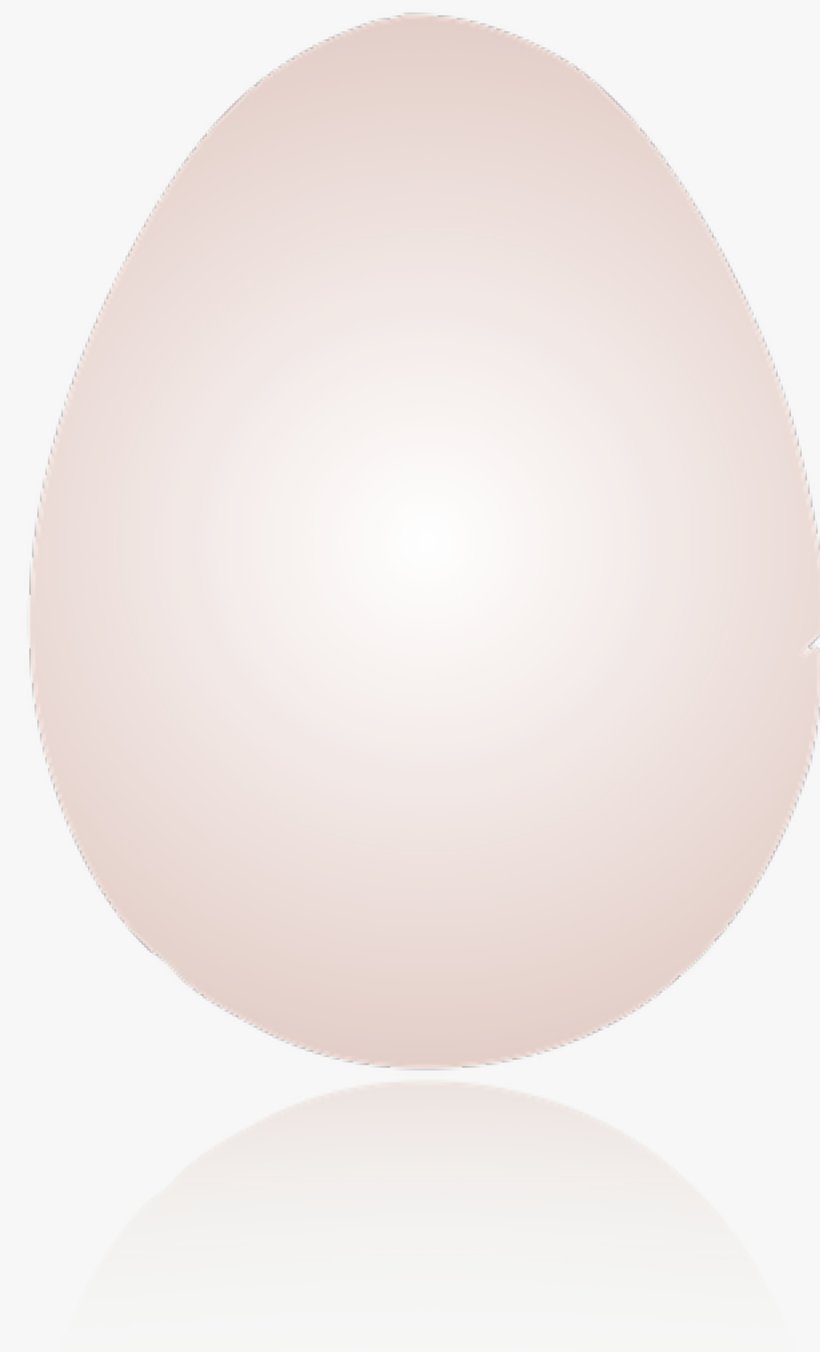


**BUILD THE GAME**



**OR**

**TWEAK THE GA**





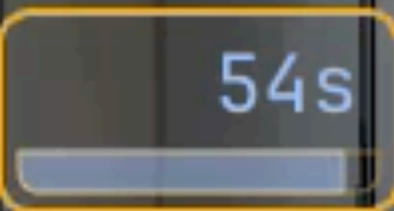




LEVEL 1

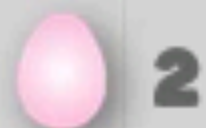


54s



SCORE

10



2



4



5

TOTAL SCORE

10



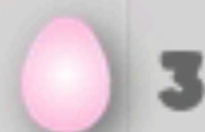


LEVEL 1

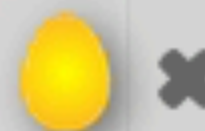


SCORE

3



3



X



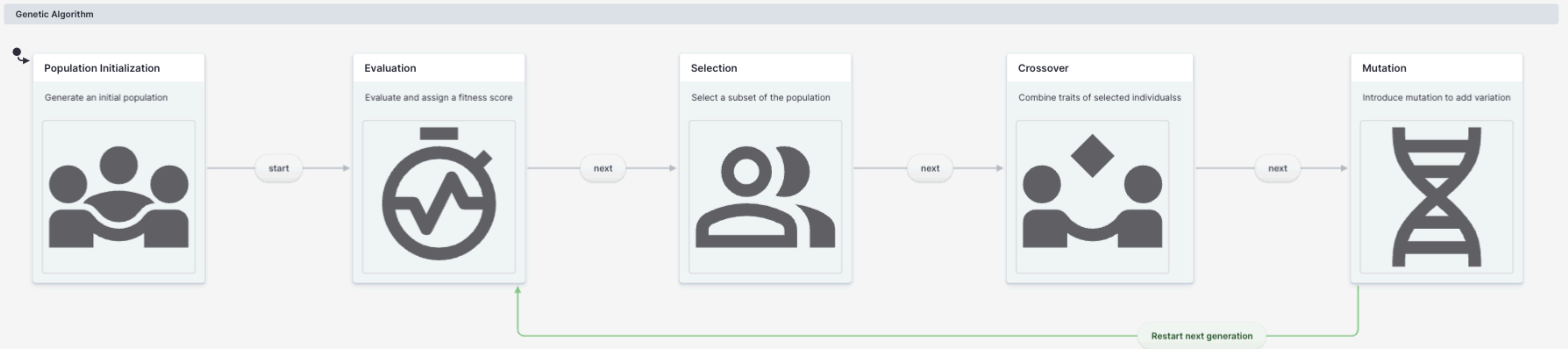
X

TOTAL  
SCORE

3



# Overcome small population size



Add genes (traits)

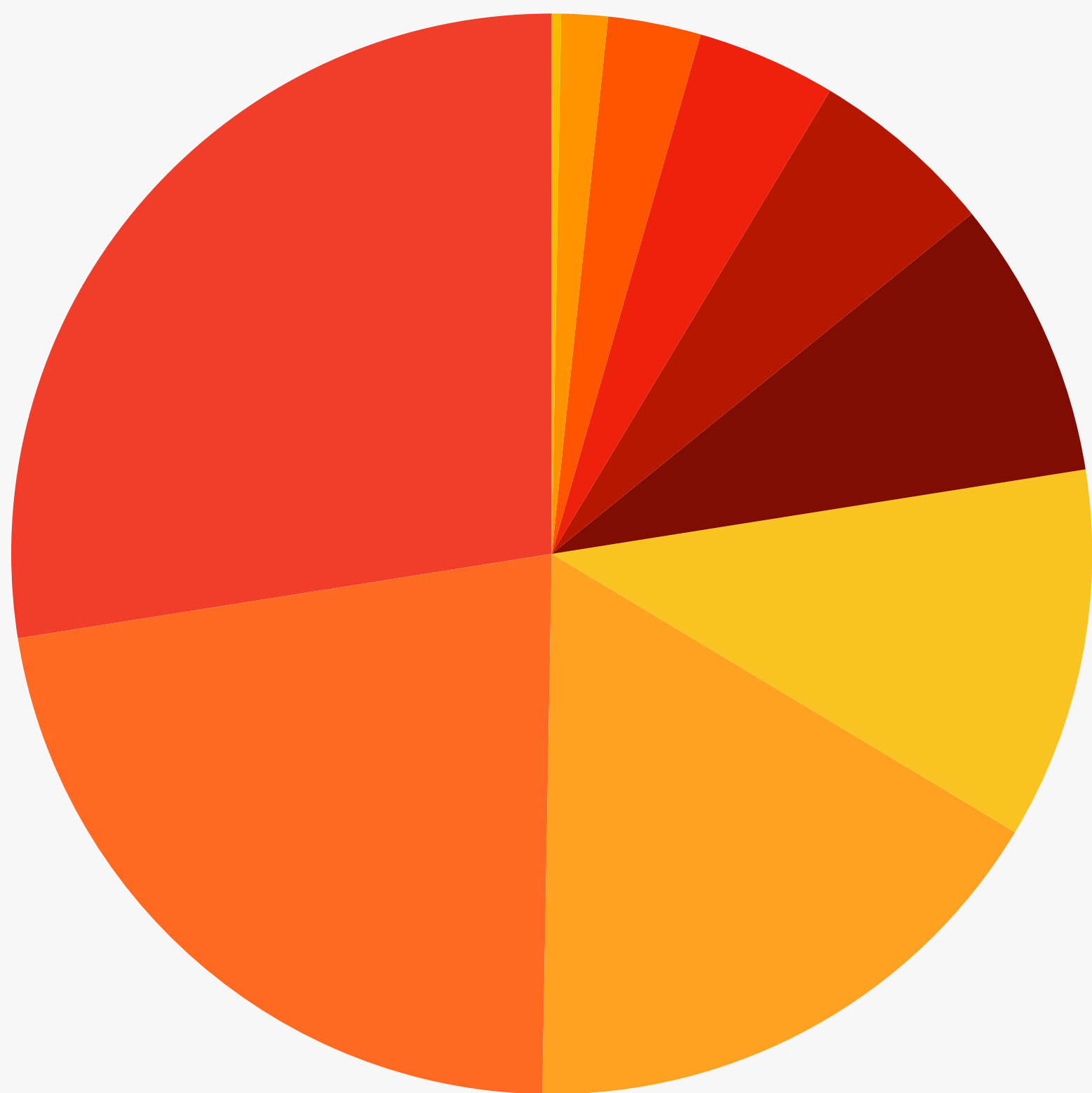
Tweak Fitness

Introduce elitism

Hybrid averaging/selection

Increase rate

# Roulette Wheel Selection



```
/**
 * Selects an individual based on their relative fitness
 * using roulette wheel selection
 * @param population
 * @returns
 */
export function rouletteWheelSelection(population: Individual[]) {
  // Calculate the total fitness of the population
  const totalFitness = population.reduce(
    (acc, individual) => acc + individual.fitness,
    0
  );

  // Generate a random number between 0 and the total fitness
  let rand = Math.random() * totalFitness;

  // Iterate through the population and select an individual based on the random
  for (let individual of population) {
    rand -= individual.fitness;
    if (rand <= 0) {
      return individual;
    }
  }

  // In case of rounding errors, return the last individual
  return population[population.length - 1];
}
```

# Individual DNA (genes)

**GENOTYPE**

0.97	0.32	0.37	0.28	0.83	0.04	0.26	0.98
------	------	------	------	------	------	------	------

**PHENOTYPE**

**{ SPEED, COLOR, SIZE, ... }**

# Individual DNA (genes)

```
export class DNA {  
  /** ...  
  static crossover(parentDNA1: DNA, parentDNA2: DNA) { ...  
  }  
  
  private id: string = '';  
  private genes: number[];  
  
  constructor(length: number) {  
    this.genes = [];  
    for (let i = 0; i < length; i++) {  
      this.genes.push(Math.random());  
    }  
  }  
}
```

## PARENT 1

0.97	0.62	0.13	0.28	0.83	0.04	0.71	0.56
------	------	------	------	------	------	------	------

## PARENT 2

0.12	0.32	0.37	0.09	0.96	0.43	0.26	0.98
------	------	------	------	------	------	------	------

# Individual DNA (genes)

```
export class DNA {  
  /** ...  
  static crossover(parentDNA1: DNA, parentDNA2: DNA) { ...  
  }  
  
  private id: string = '';  
  private genes: number[];  
  
  constructor(length: number) {  
    this.genes = [];  
    for (let i = 0; i < length; i++) {  
      this.genes.push(Math.random());  
    }  
  }  
}
```

## PARENT 1



## PARENT 2



# Individual DNA (genes)

```
export class DNA {  
  /** ...  
  static crossover(parentDNA1: DNA, parentDNA2: DNA) { ...  
  }  
  
  private id: string = '';  
  private genes: number[];  
  
  constructor(length: number) {  
    this.genes = [];  
    for (let i = 0; i < length; i++) {  
      this.genes.push(Math.random());  
    }  
  }  
}
```

**AGNOSTIC OF PHENOTYPE VALUES**



## PARENT 1



## PARENT 2



## CHILD





```
export type PhenotypeKey =  
  | 'speed'  
  | 'baseTweenDurationSeconds'  
  | 'stationaryEggLayingRate'  
  | 'movingEggLayingRate'  
  | 'hatchRate'  
  | 'minXMovement'  
  | 'maxXMovement'  
  | 'minStopMS'  
  | 'maxStopMS'  
  | 'maxEggs'  
  | 'blackEggRate'  
  | 'goldEggRate'  
  | 'restAfterLayingEggMS';
```

```
export const phenotypeConfig: PhenotypeConfig = {  
  // The x speed of the hen  
  speed: {  
    min: 0,  
    max: 1,  
  },  
  // The maximum number of eggs the hen can lay  
  maxEggs: {  
    min: 1,  
    max: 10,  
    round: true,  
  },  
  // The likelihood of the hen laying an egg while stationary  
  stationaryEggLayingRate: {  
    min: 0,  
    max: 0.5,  
  },  
  // The likelihood of the hen laying an egg while moving  
  movingEggLayingRate: {  
    min: 0,  
    max: 0.5,  
  },  
}
```

```
// The min x amount a hen can move during its animation
```

```
export type PhenotypeKey =  
  | 'speed'  
  | 'baseTweenDurationSeconds'  
  | 'stationaryEggLayingRate'  
  | 'movingEggLayingRate'  
  | 'hatchRate'  
  | 'minXMovement'  
  | 'maxXMovement'  
  | 'minStopMS'  
  | 'maxStopMS'  
  | 'maxEggs'  
  | 'blackEggRate'  
  | 'goldEggRate'  
  | 'restAfterLayingEggMS';
```

```
export const phenotypeConfig: PhenotypeConfig = {  
  // The x speed of the hen  
  speed: {  
    min: 0,  
    max: 1,  
  },  
  // The maximum number of eggs the hen can lay  
  maxEggs: {  
    min: 1,  
    max: 10,  
    round: true,  
  },  
  // The likelihood of the hen laying an egg while stationary  
  stationaryEggLayingRate: {  
    min: 0,  
    max: 0.5,  
  },  
  // The likelihood of the hen laying an egg while moving  
  movingEggLayingRate: {  
    min: 0,  
    max: 0.5,  
  },  
}
```

## CHILD

0.97

0.32

0.37

0.28

0.83

0.04

0.26

0.98

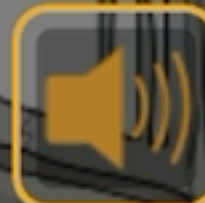
```
/** Genetic Algorithm individual of the population */  
export interface Individual {  
  dna: DNA;  
  phenotype: Record<PhenotypeKey, number>;  
  fitness: number;  
}
```



```
/** Hendividual = Hen + Individual for Egg Drop */  
export interface Hendividual extends Individual {  
  id: string;  
  
  // Configuration  
  initialPosition: Position;  
  
  // Results  
  stats: {  
    eggsLaid: number;  
    eggsCaught: {  
      white: number;  
      gold: number;  
      black: number;  
    };  
    eggsHatched: number;  
    eggsBroken: number;  
    eggsOffscreen: number;  
  };  
}
```



LEVEL 3



36

**SCORE**  
**39**











16  
 3  
 2

**TOTAL SCORE**  
**93**





Test sound

Generation	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
totalEggsLaid	68	76	96	88	95	79	72	57	73	75	80	67	73	68	60	73
averageEggsLaid	1.7	1.9	2.4	2.2	2.4	2.0	1.8	1.4	1.8	1.9	2.0	1.7	1.8	1.7	1.5	1.8
catchRate	35%	38%	30%	27%	36%	33%	39%	37%	27%	28%	29%	36%	38%	21%	25%	29%
averageFitness	0.78	0.89	1.03	0.95	0.95	0.90	0.88	0.79	0.90	0.94	1.02	0.84	0.83	1.06	0.88	0.92
averageHenSpeed 	0.51	0.71	0.74	0.75	0.81	0.80	0.85	0.91	0.94	0.95	0.96	0.96	0.98	0.98	0.98	0.98
averageBaseTweenDurationSeconds	3	3	4	4	4	4	4	5	5	5	5	5	5	5	5	5
averageStationaryEggLayingRate 	0.25	0.36	0.37	0.38	0.40	0.40	0.43	0.46	0.47	0.48	0.48	0.48	0.49	0.49	0.49	0.49
averageMovingEggLayingRate 	0.25	0.36	0.37	0.38	0.40	0.40	0.43	0.46	0.47	0.48	0.48	0.48	0.49	0.49	0.49	0.49
averageHatchRate	0.5	0.7	0.7	0.8	0.8	0.8	0.9	0.9	0.9	1.0	1.0	1.0	1.0	1.0	1.0	1.0
averageMinXMovement 	126	156	160	163	171	170	178	187	191	193	194	195	197	197	197	197
averageMaxXMovement 	771	982	1,009	1,025	1,084	1,076	1,127	1,188	1,219	1,229	1,235	1,242	1,258	1,258	1,258	1,258
averageMinStopMS 	506	711	737	753	810	802	852	911	941	951	956	964	979	979	979	979
averageMaxStopMS 	2,532	3,557	3,684	3,762	4,049	4,011	4,259	4,552	4,704	4,755	4,780	4,818	4,894	4,894	4,894	4,894
averageMaxEggs 	5.5	7.4	7.6	7.8	8.4	8.3	8.8	9.3	9.6	9.7	9.8	9.9	10.0	10.0	10.0	10.0
averageBlackEggRate 	0.15	0.21	0.22	0.23	0.24	0.24	0.26	0.27	0.28	0.29	0.29	0.29	0.29	0.29	0.29	0.29
averageGoldEggRate 	0.25	0.36	0.37	0.38	0.40	0.40	0.43	0.46	0.47	0.48	0.48	0.48	0.49	0.49	0.49	0.49
averageRestAfterLayingEggMS	1,013	1,423	1,474	1,505	1,620	1,604	1,704	1,821	1,882	1,902	1,912	1,928	1,958	1,958	1,958	1,958

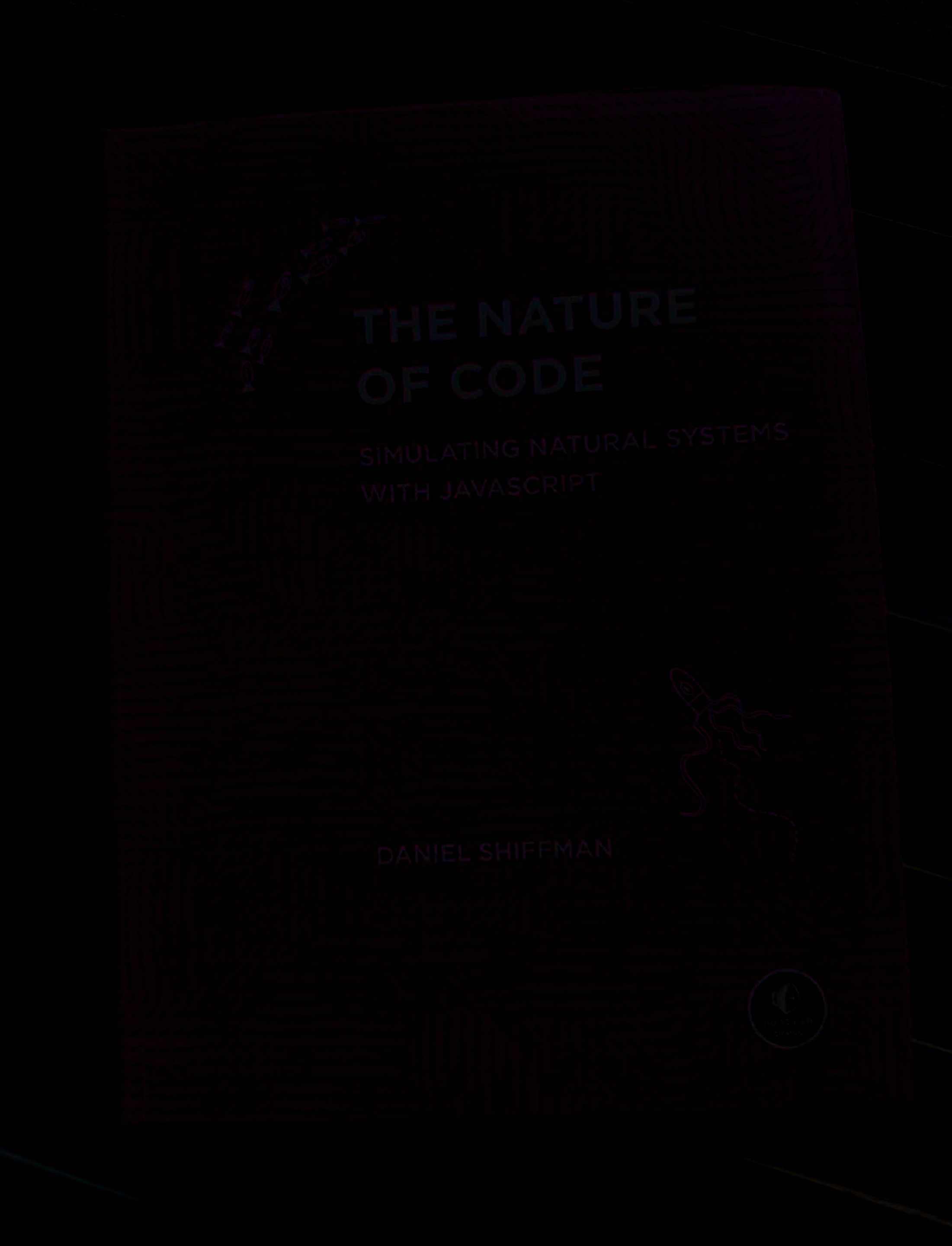
Test sound

Generation	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
totalEggsLaid	68	76	96	88	95	79	72	57	73	75	80	67	73	68	60	73
averageEggsLaid	1.7	1.9	2.4	2.2	2.4	2.0	1.8	1.4	1.8	1.9	2.0	1.7	1.8	1.7	1.5	1.8
catchRate	35%	38%	30%	27%	36%	33%	39%	37%	27%	28%	29%	36%	38%	21%	25%	29%
averageFitness	0.78	0.89	1.03	0.95	0.95	0.90	0.88	0.79	0.90	0.94	1.02	0.84	0.83	1.06	0.88	0.92
averageHenSpeed	0.51	0.71	0.74	0.75	0.81	0.80	0.85	0.91	0.94	0.95	0.96	0.96	0.98	0.98	0.98	0.98
averageBaseTweenDurationSeconds	3	3	4	4	4	4	4	5	5	5	5	5	5	5	5	5
averageStationaryEggLayingRate	0.25	0.36	0.37	0.38	0.40	0.40	0.43	0.46	0.47	0.48	0.48	0.48	0.49	0.49	0.49	0.49
averageMovingEggLayingRate	0.25	0.36	0.37	0.38	0.40	0.40	0.43	0.46	0.47	0.48	0.48	0.48	0.49	0.49	0.49	0.49
averageHatchRate	0.5	0.7	0.7	0.8	0.8	0.8	0.9	0.9	0.9	1.0	1.0	1.0	1.0	1.0	1.0	1.0
averageMinXMovement	126	156	160	163	171	170	178	187	191	193	194	195	197	197	197	197
averageMaxXMovement	771	982	1,009	1,025	1,084	1,076	1,127	1,188	1,219	1,229	1,235	1,242	1,258	1,258	1,258	1,258
averageMinStopMS	506	711	737	753	810	802	852	911	941	951	956	964	979	979	979	979
averageMaxStopMS	2,532	3,557	3,684	3,762	4,049	4,011	4,259	4,552	4,704	4,755	4,780	4,818	4,894	4,894	4,894	4,894
averageMaxEggs	5.5	7.4	7.6	7.8	8.4	8.3	8.8	9.3	9.6	9.7	9.8	9.9	10.0	10.0	10.0	10.0
averageBlackEggRate	0.15	0.21	0.22	0.23	0.24	0.24	0.26	0.27	0.28	0.29	0.29	0.29	0.29	0.29	0.29	0.29
averageGoldEggRate	0.25	0.36	0.37	0.38	0.40	0.40	0.43	0.46	0.47	0.48	0.48	0.48	0.49	0.49	0.49	0.49
averageRestAfterLayingEggMS	1,013	1,423	1,474	1,505	1,620	1,604	1,704	1,821	1,882	1,902	1,912	1,928	1,958	1,958	1,958	1,958

# **SUMMARY**

- 1. GENETIC ALGORITHMS**
- 2. HOW TO CREATE A GAME**
- 3. HOW TO INCLUDE A GA**





**GET THIS BOOK!**

**The Nature of Code**

Daniel Shiffman

[thecodingtrain.com/](http://thecodingtrain.com/)





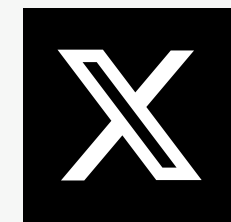
**Kevin Maes**



**STATELY**



[linkedin.com/in/kevinmaes](https://www.linkedin.com/in/kevinmaes)



[@kvmaes](https://twitter.com/kvmaes)



[github.com/kevinmaes](https://github.com/kevinmaes)