

PaSe: An Extensible and Inspectable DSL for Micro-Animations

Ruben Pieters & Tom Schrijvers

7 Best Animation Libraries for UI Designers 2018 | Codementor
<https://www.codementor.io/hayeskier/7-best-animation-libraries-for-ui-designers-2018>
Jun 29, 2018 · Overall, a great **animation library** for UI designers in 2018. A must-try. AnimeJS. GitHub. AnimeJS is fairly new JS **animation library** compared to others listed here. In the small span of its release, it has grown rapidly and also shows promise to become of the **best** out there. AnimeJS is a complete package when it comes to **animation library**.

The 10 best JavaScript libraries for SVG animation
<https://noeticforce.com/javascript-libraries-for-svg-animation>

Sep 15, 2019 · The 10 **Best JavaScript Libraries for SVG Animation** Velocity, Walkway, RaphaelJS, Snap.Svg, Bonsai, Lazy Line Painter, Vivus, Progre
Conclusion.

Top 20 jQuery Animation Library and Plugin
<https://colorlib.com/wp/jquery-animation-library/>

Mar 28, 2019 · Top 20 jQuery **Animation Library**
Powered by mo.js. Motion Graphics for the W
Animation. Material Scroll **Animation**. Elar
pageSwitch for JavaScript. Animating

15 Best JavaScript Anima
<https://codegeekz.com/15-best-j>

For this round-up, we would like to intru
of jQuery when it comes to **animation** effe
1. Tween JS. TweenJS is a simple tweening li
to integrate well with the EaseJS **library**, but is

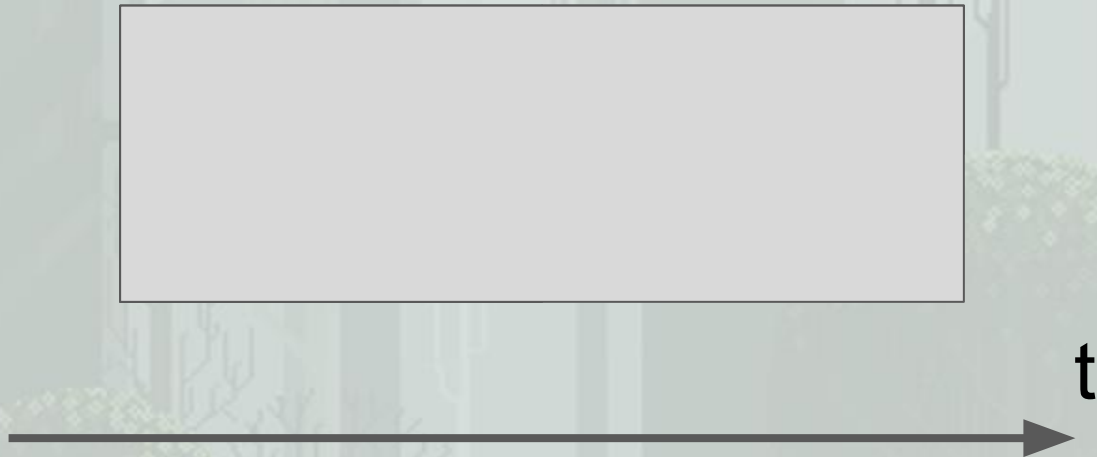
9 of the Best Animation Libraries for UI Designers — SitePoint
<https://www.sitepoint.com/our-top-9-animation-libraries/>
Animation is a part of a UI designer's job. Here are 9 free **animation libraries** we think deliver the most power for the smallest file size, while being relatively easy to use for reasonably code ...

10 Best Free Animation Libraries For The Web | Webdesigner Depot
<https://www.webdesignerdepot.com/2018/01/10-best-free-animation-libraries-for-the-web/>
Jan 22, 2018 · **Animation** is one of the key trends for 2018, but you don't have to reinvent the wheel to add movement to your sites. We look at the best free **animation libraries** for web projects. You can build some crazy things with UI **animations** on the web. This is quickly becoming the norm for modern websites ...

The Animation Library | Your Partner for Quality Animations
<https://www.theanimationlibrary.com>
Animating has never been so easy. We offer a wide variety of pre-rigged Characters and Custom services for your project. Discover The **Animation Library** today.

CSS Animation Libraries | CSS-Tricks
<https://css-tricks.com/css-animation-libraries/>
Jul 22, 2019 · There are an awful lot of libraries that want to help you animate things on the web. These aren't really libraries that help you with the syntax or the technology of **animations**, but rather are grab-and-use as-is libraries.

- Express Animations Compositionally
- Give a Step-By-Step Introduction to PaSe
- Taster of Inspectability of DSLs

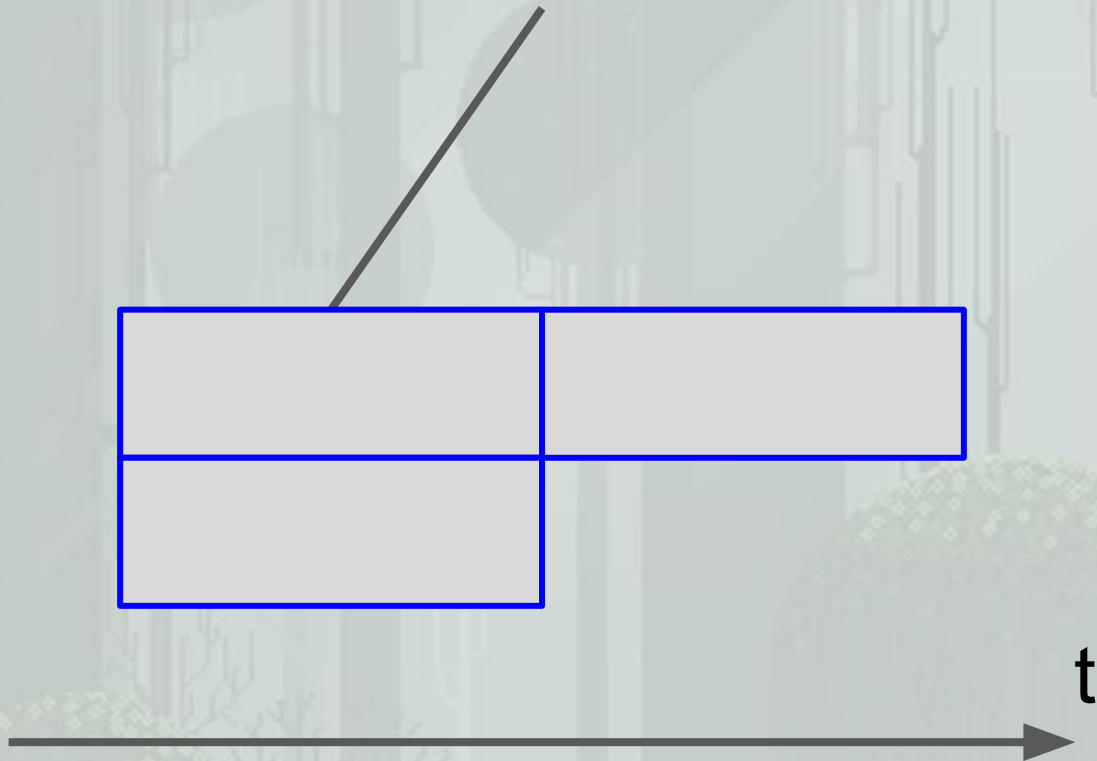




Background - edermunizz (itch.io)
Adventurer, Slime - rvros (itch.io)

Overview

Sequential
Automata



Parallel Sequential



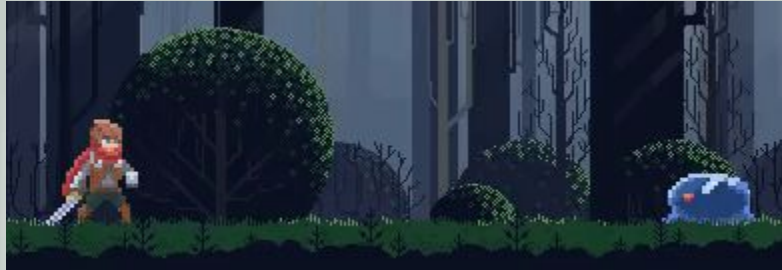
linearTo





```
moveX =  
  linearTo (player . x) (For 1) (To 300)
```

Property Duration Target

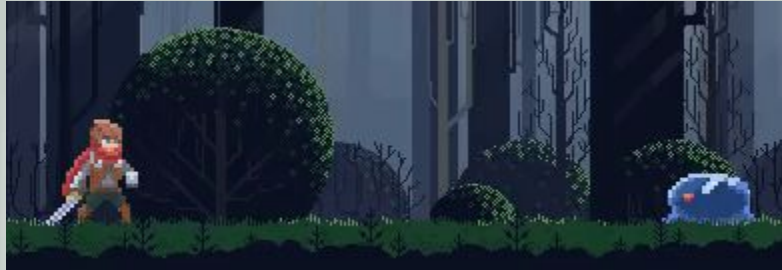


moveX

moveSheet

t





```
movePlayer =  
  moveX `parallel` moveSheet
```



swing

minusOne

t



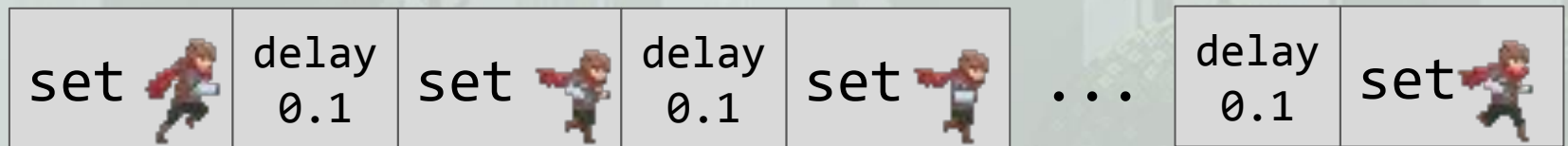


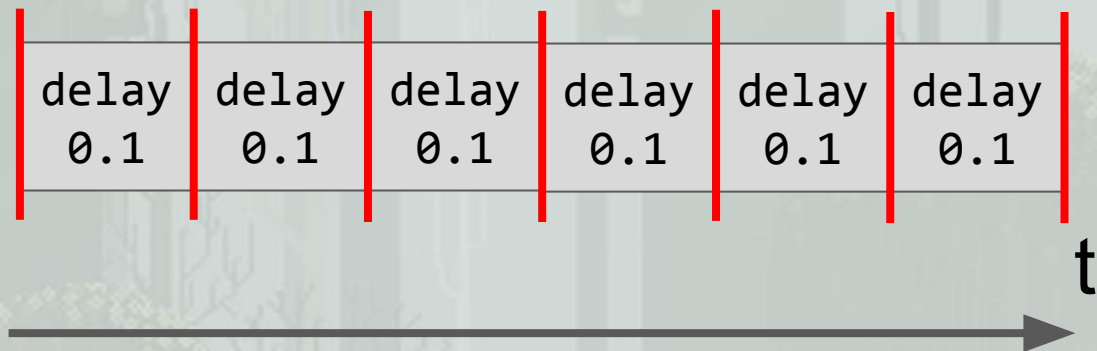
```
hit =  
    swing `sequential` minusOne
```

Encoding Animation Techniques

Frame by Frame Animation









```
moveSheet =  
  set (player . texture) "frame0"  
  `sequential`  
  delay 0.1  
  `sequential`  
  set (player . texture) "frame1"  
  `sequential`  
  ...  
  `sequential`  
  delay 0.1  
  `sequential`  
  set (player . texture) "frame5"
```

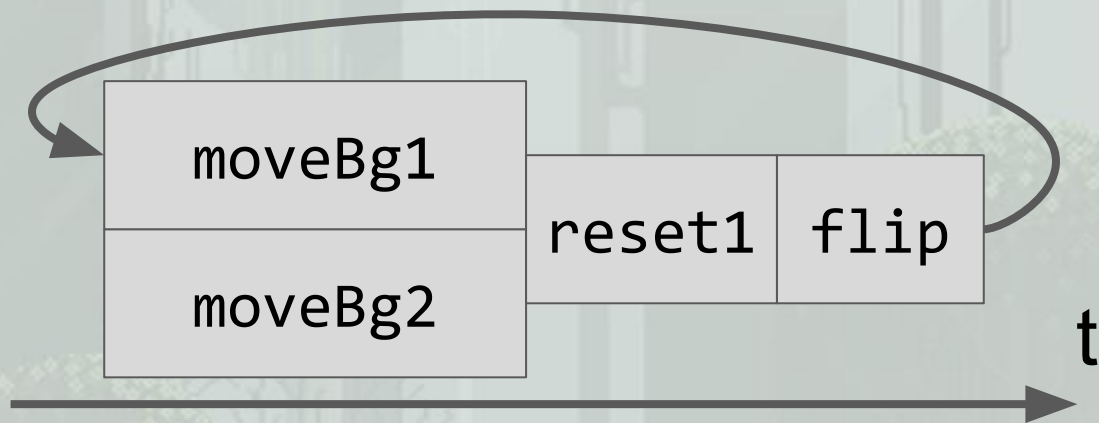
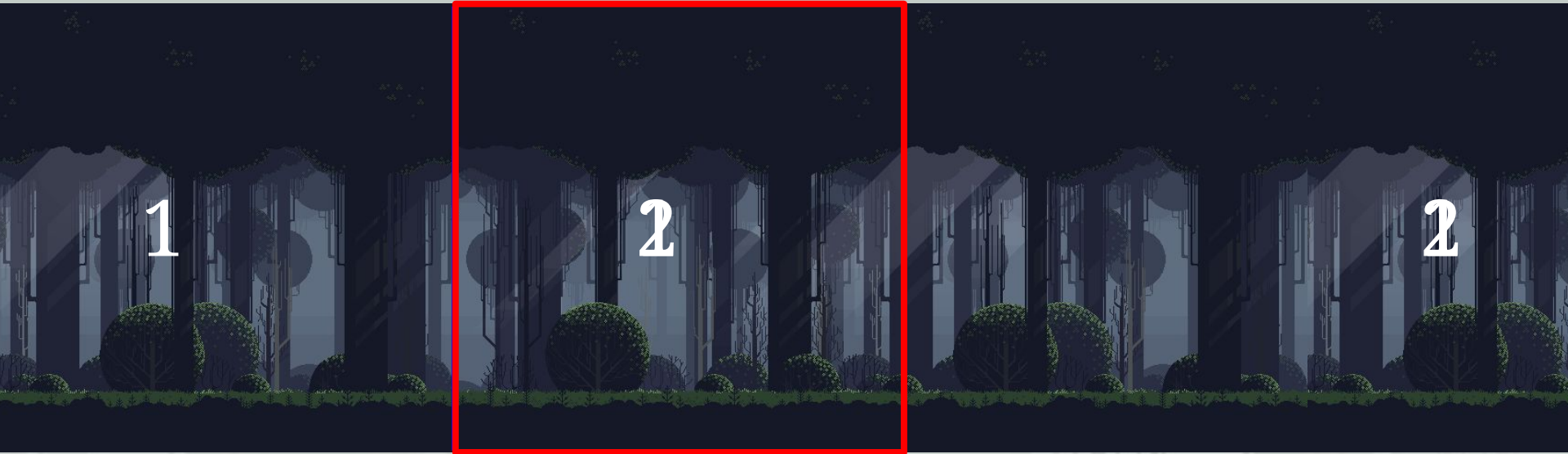


```
moveSheet = frameByFrame (player . texture) 0.1  
  ["frame0", ..., "frame5"]
```

```
frameByFrame lens delay list = ...
```

Scrolling Background





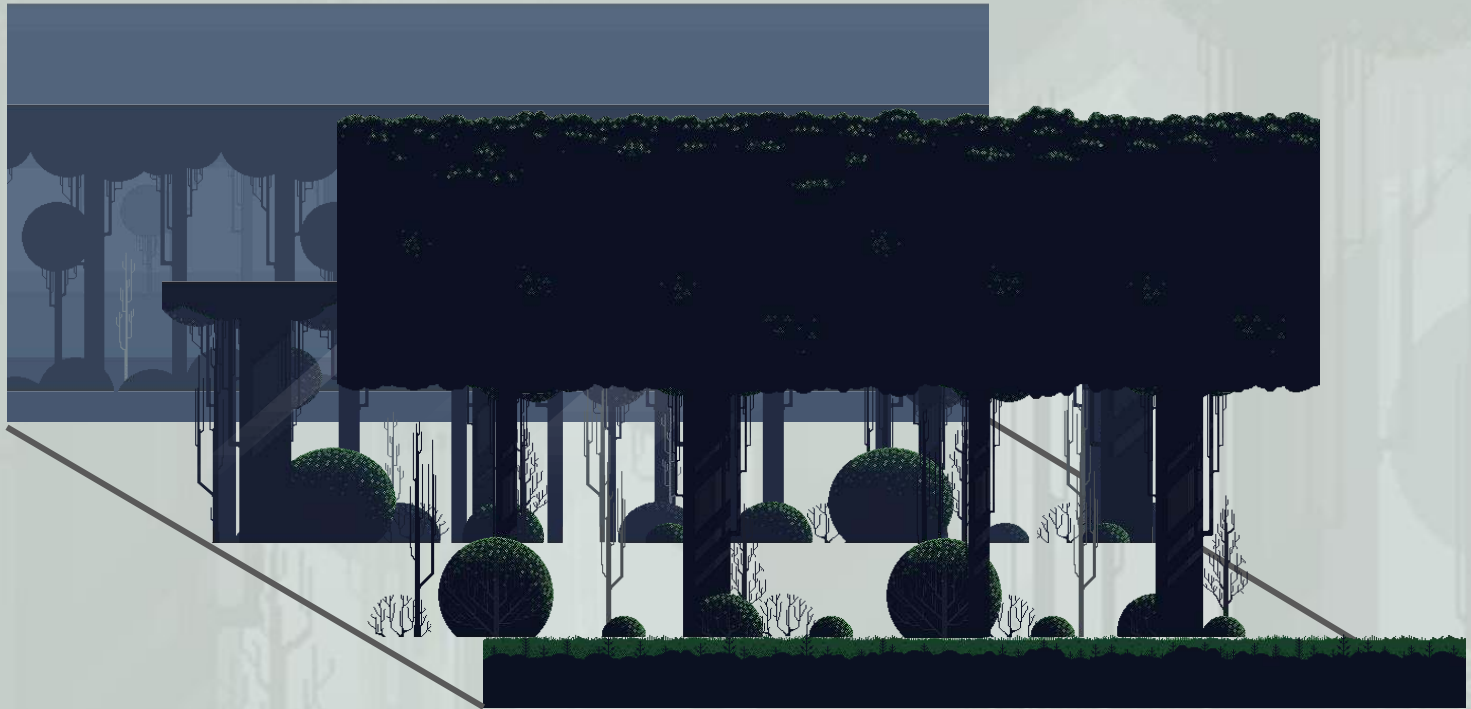


```
scrollingBg dur bg1 bg2 =  
  (moveBg dur bg1 `parallel` moveBg dur bg2)  
  `sequential`  
  set (bg1 . x) 0  
  `sequential`  
  scrollingBg dur bg2 bg1
```

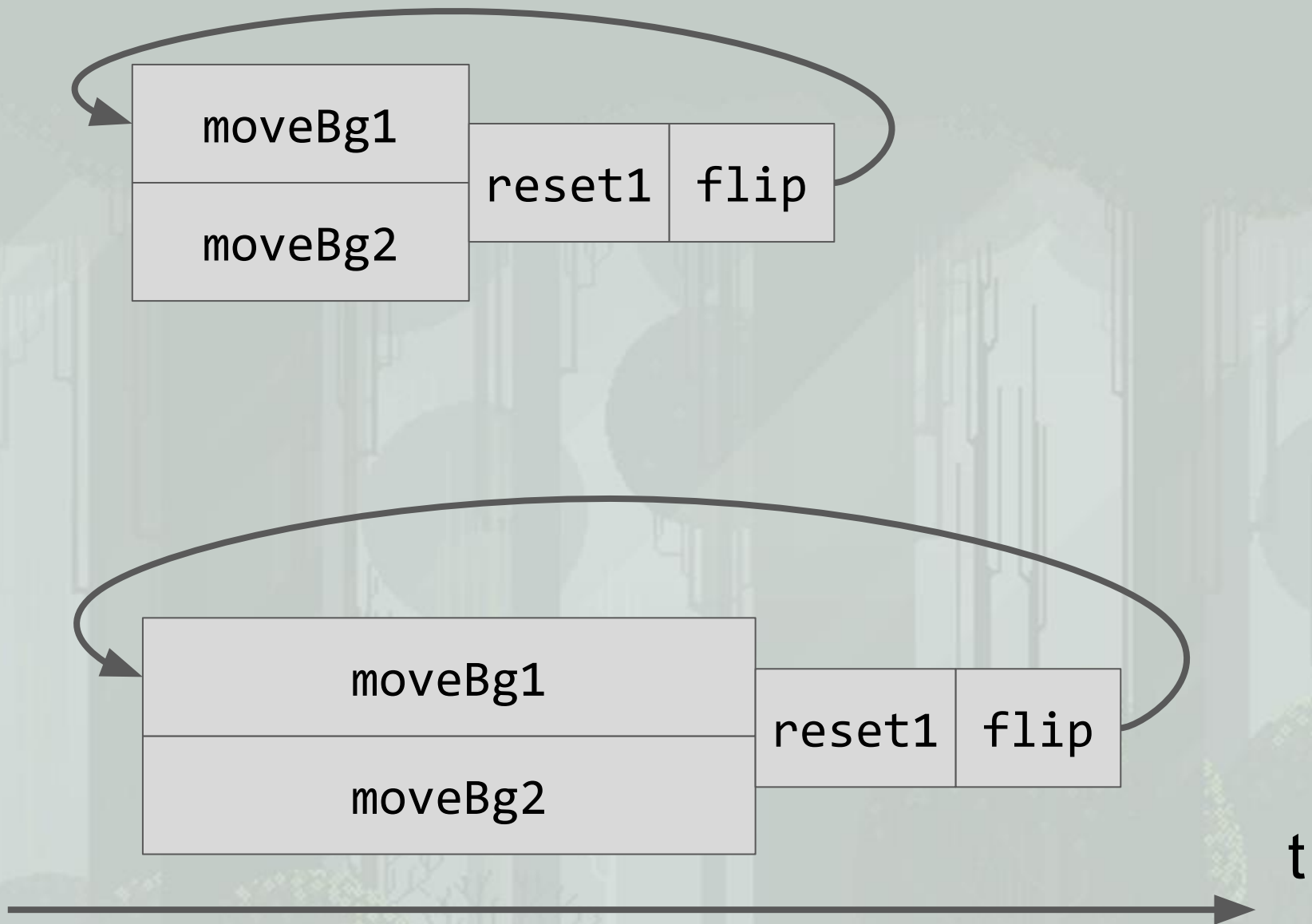
Parallax

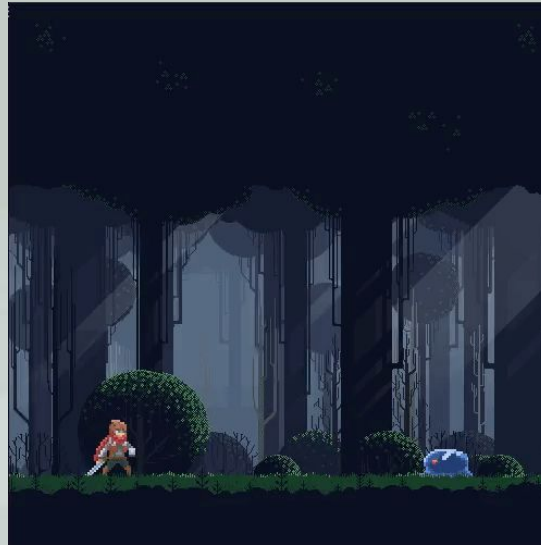


slow

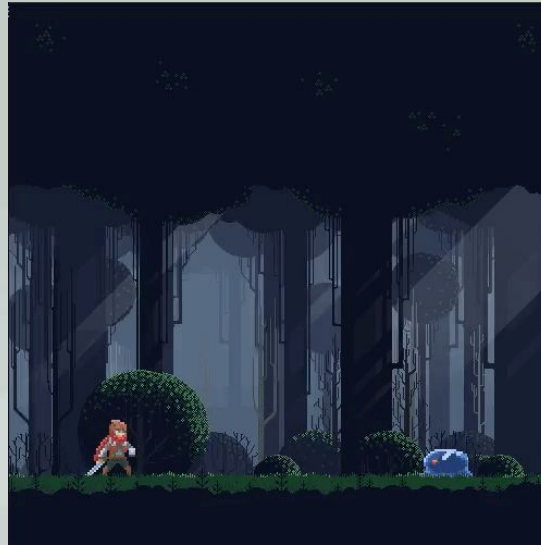


fast





```
parallaxBg =  
  scrollingBg 5 bg1A bg1B  
  `parallel`  
  scrollingBg 7.5 bg2A bg2B  
  `parallel`  
  scrollingBg 10 bg3A bg3B  
  `parallel`  
  scrollingBg 15 bg4A bg4B
```



```
parallaxBg = parallax  
[ (5, bg1A, bg1B)  
  , (7.5, bg2A, bg2B)  
  , (10, bg3A, bg3B)  
  , (15, bg4A, bg4B)  
]
```

```
parallax = ...
```

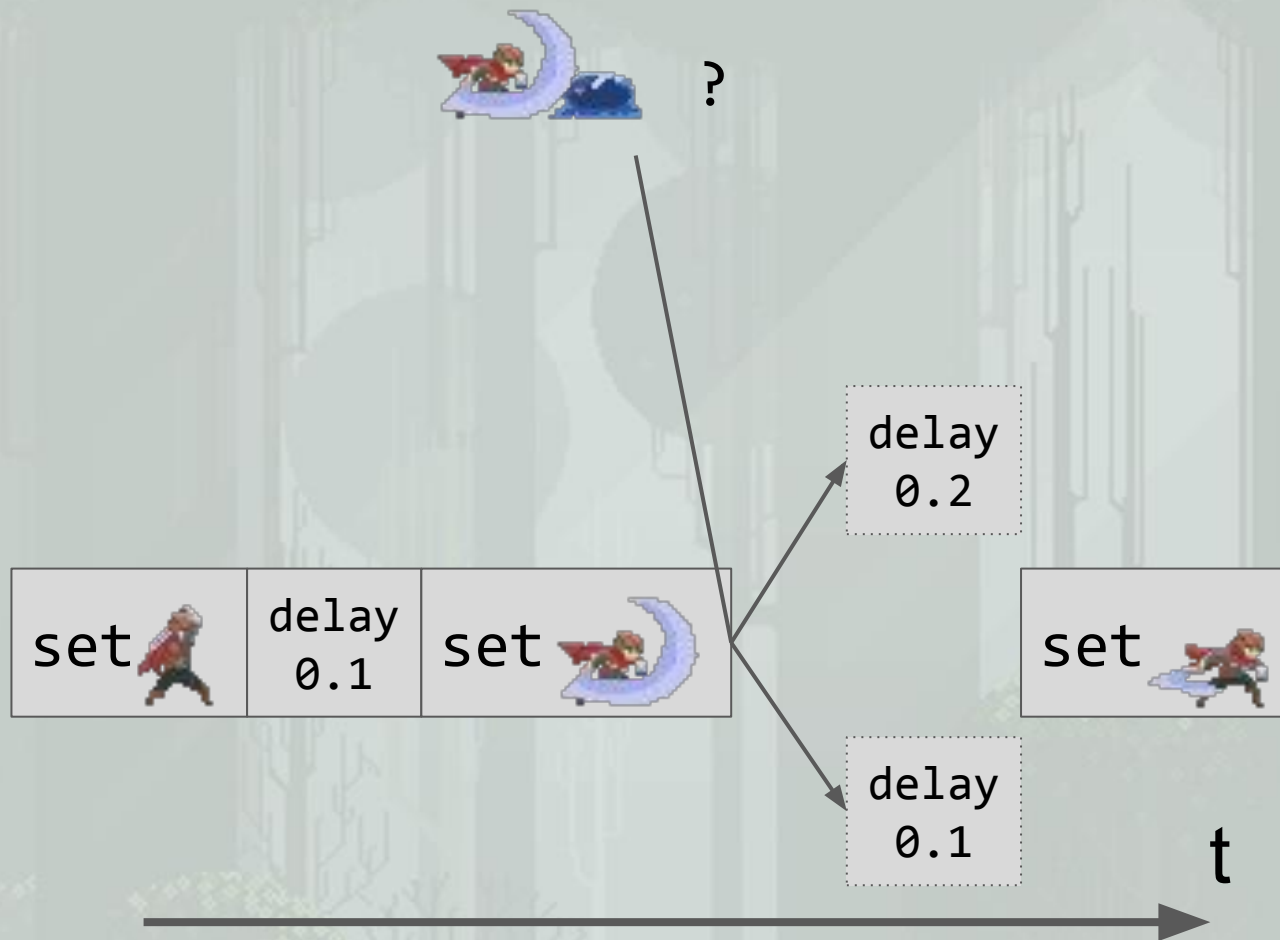
Hit Stop





t





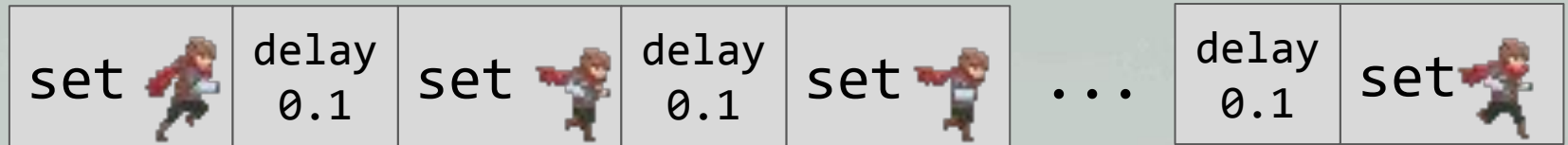


delay
0.2

delay
0.1

```
hitStop =  
  ifThenElse condition (delay 0.2) (delay 0.1)
```

Inspectability



textures =

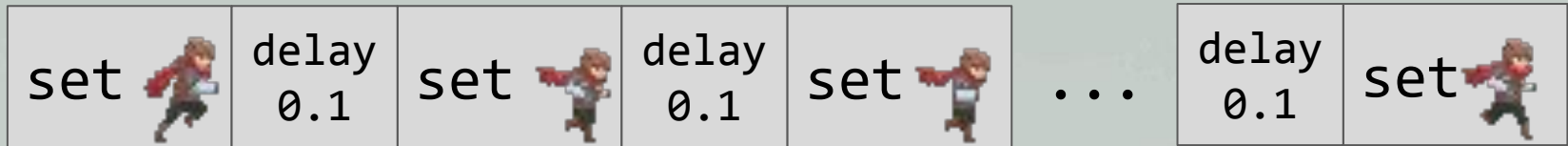


duration =

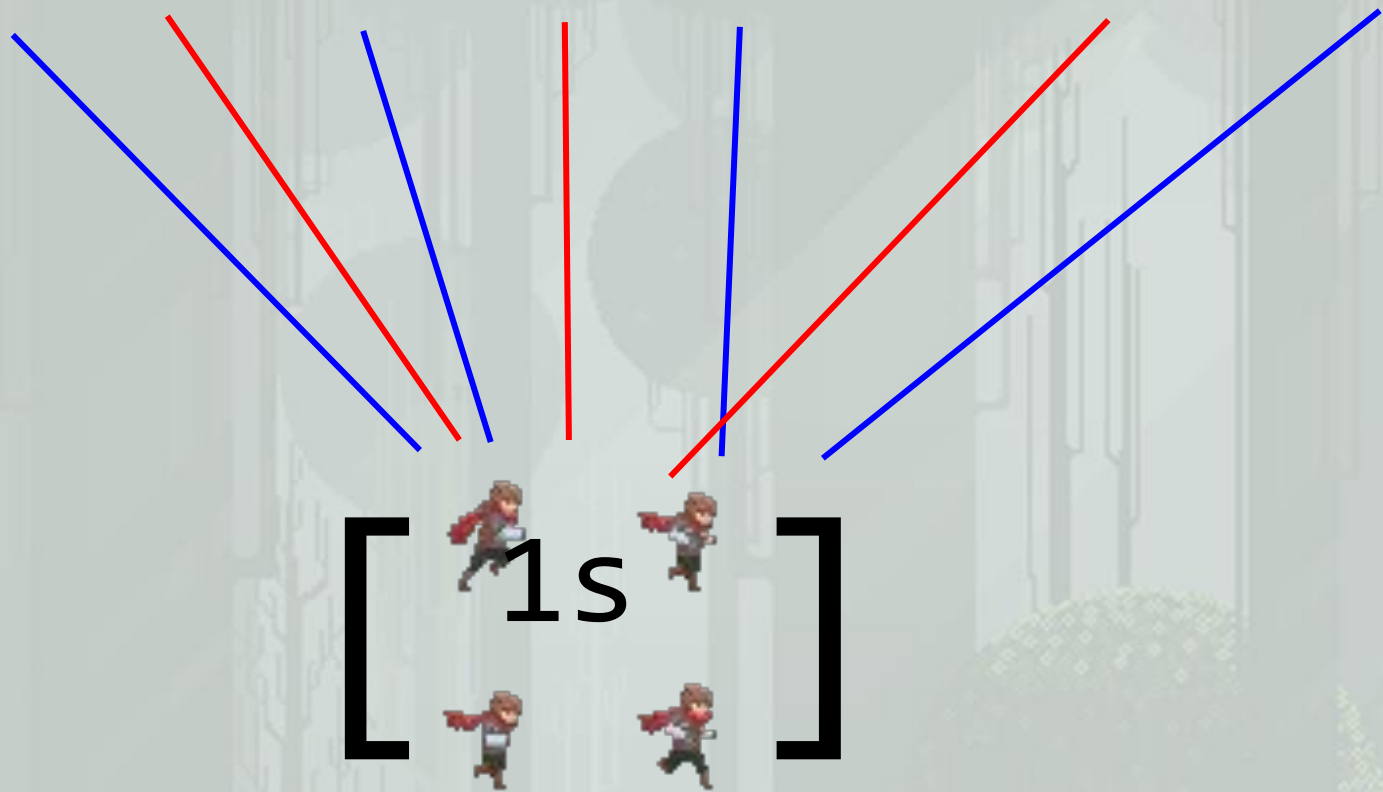
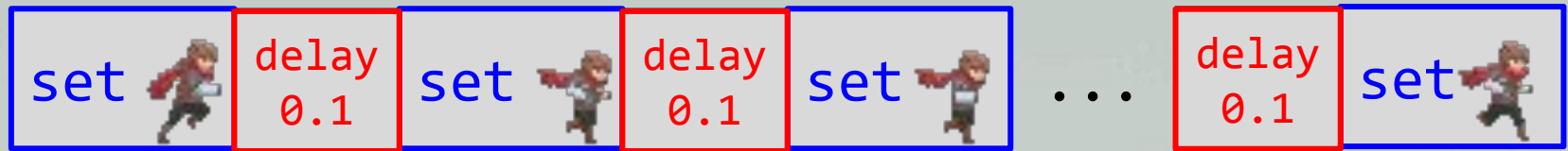
1s

noConflict =

ok



```
allTextures =
[ "playerIdle.png"
  , "playerAttack0.png"
  , "playerAttack1.png"
  , "playerAttack2.png"
  , "playerAttack3.png"
  ...
]
```



anim a

a -> anim b

Expressivity



Inspectability

Conclusion

- Expressing Animations Compositionally
- Step-By-Step Introduction to PaSe
- Taster of Inspectability of DSLs

Paper

PaSe: An Extensible and Inspectable DSL for Micro-Animations

Ruben P. Pieters ^[0000-0003-0537-9403] and Tom Schrijvers^[0000-0001-8771-5559]

KU Leuven, 3001 Leuven, Belgium
{ruben.pieters, tom.schrijvers}@cs.kuleuven.be

Abstract. This paper presents PaSe, an extensible and inspectable DSL embedded in Haskell for expressing micro-animations. The philosophy of PaSe is to compose animations based on sequential and parallel composition of smaller animations. This differs from other animation libraries that focus more on sequential composition and have only limited forms of parallel composition. To provide similar flexibility as other animation libraries, PaSe features extensibility of operations and inspectability of animations. We present the features of PaSe with a to-do list application, discuss the PaSe implementation, and argue that the callback style of extensibility is detrimental for correctly combining PaSe features. We contrast with the GreenSock Animation Platform, a professional-grade and widely used JavaScript animation library, to illustrate this point.

1 Introduction

Monads quickly became ubiquitous in functional programming because of their ability to structure effectful code in a pure functional codebase [22]. However, monads have two major drawbacks. First, monads are not trivially extensible. A variety of techniques were developed to resolve this, including monad transformers [14], free monads, and algebraic effects and handlers [21]. Second, monadic computations can only be inspected up to the next action. Techniques such as applicative functors [16], arrows [9], or selective applicative functors [18] increase the inspection capabilities by reducing the expressivity compared to monads.

This paper develops a domain specific language (DSL) embedded in Haskell for defining micro-animations, called PaSe¹. PaSe employs the aforementioned techniques to support its key features: extensibility of operations and inspectability of animations while providing the freedom to express arbitrary animations.

Micro-animations are short animations displayed when users interact with an

- DSL Design using MTL/Finally
- Tagless encoding
- More Detailed Exposition of Expressivity/Inspectability Trade-off

PaSe: An Extensible and Inspectable DSL for Micro-Animations

Ruben P. Pieters [0000-0003-0537-9403] and Tom
Schrijvers [0000-0001-8771-5559]

KU Leuven, 3001 Leuven, Belgium
{ruben.pieters, tom.schrijvers}@cs.kuleuven.be

Abstract. This paper presents PaSe, an extensible and inspectable DSL embedded in Haskell for expressing micro-animations. The philosophy of PaSe is to compose animations based on sequential and parallel composition of smaller animations. This differs from other animation libraries that focus more on sequential composition and have only limited forms of parallel composition. To provide similar flexibility as other animation libraries, PaSe features extensibility of operations and inspectability of animations. We present the features of PaSe with a to-do list application, discuss the PaSe implementation, and argue that the callback style of extensibility is detrimental for correctly combining PaSe features. We contrast with the GreenSock Animation Platform, a professional-grade and widely used JavaScript animation library, to illustrate this point.

1 Introduction

Monads quickly became ubiquitous in functional programming because of their ability to structure effectful code in a pure functional codebase [22]. However, monads have two major drawbacks. First, monads are not trivially extensible. A variety of techniques were developed to resolve this, including monad transformers [14], free monads, and algebraic effects and handlers [21]. Second, monadic computations can only be inspected up to the next action. Techniques such as applicative functors [16], arrows [9], or selective applicative functors [18] increase the inspection capabilities by reducing the expressivity compared to monads.

This paper develops a domain specific language (DSL) embedded in Haskell for defining micro-animations, called PaSe¹. PaSe employs the aforementioned techniques to support its key features: extensibility of operations and inspectability of animations while providing the freedom to express arbitrary animations.

Micro-animations are short animations displayed when users interact with an



github.com/rubenpieters/anim_eff_dsl



arxiv.org/abs/2002.02171



github.com/rubenpieters/PaSe-hs



ruben.pieters@cs.kuleuven.be



Particles







```
particle = do  
  id <- create  
  (moveUp id `parallel` fadeOut id)  
  delete id
```



withParticle **-1**

moveUp

fadeOut

t



```
particle =  
  withParticle $ \id ->  
    moveUp id `parallel` fadeOut id
```