

Polyphony: Programming Interfaces and Interactions with the Entity-Component-System Model

Thibault Raffailac, Stéphane Huot



Programming UIs with ECS

1. ECS, a composition model for video games
2. Polyphony, an experimental interaction toolkit
3. Designing UIs with *composition over inheritance*
4. Contributions, and future work

Entity-Component-System

An architectural pattern developed for video games

Alternative to Object-Oriented Programming

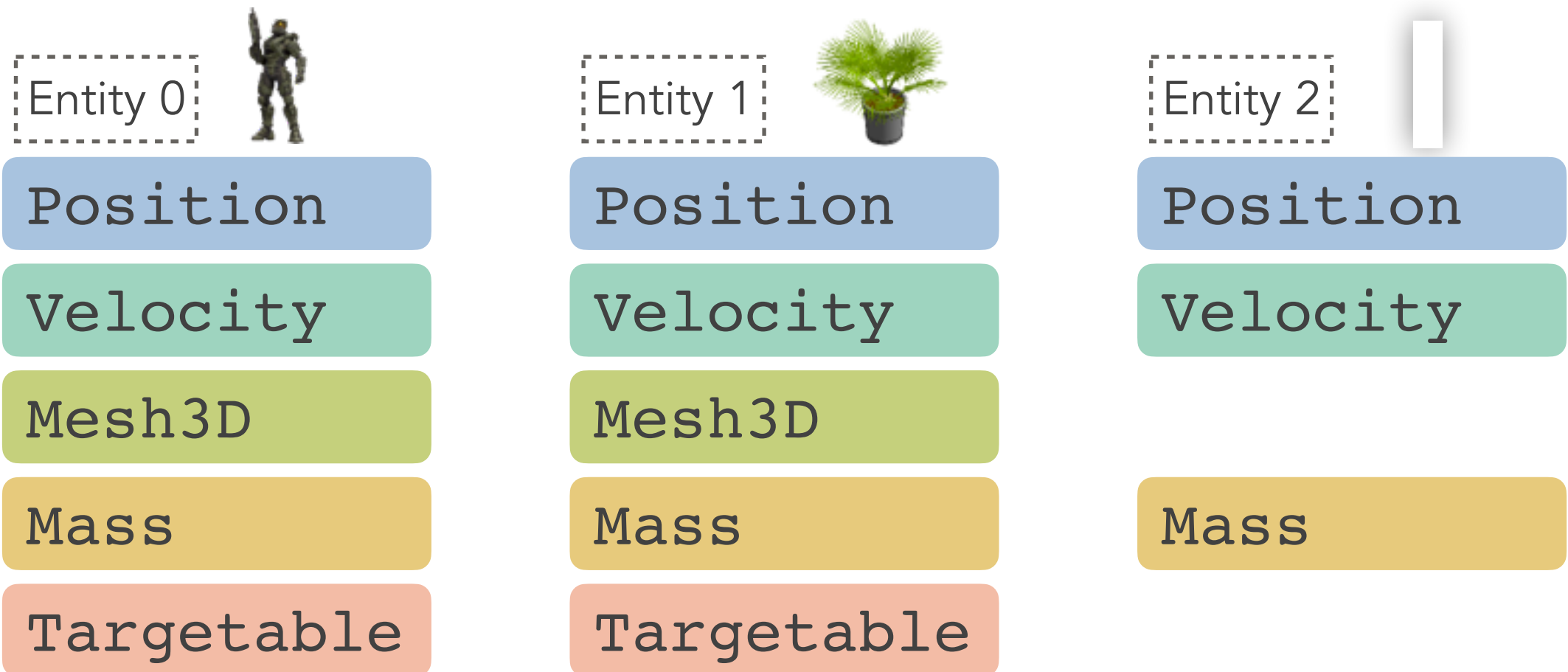
Entity-Component-System

Entity 0

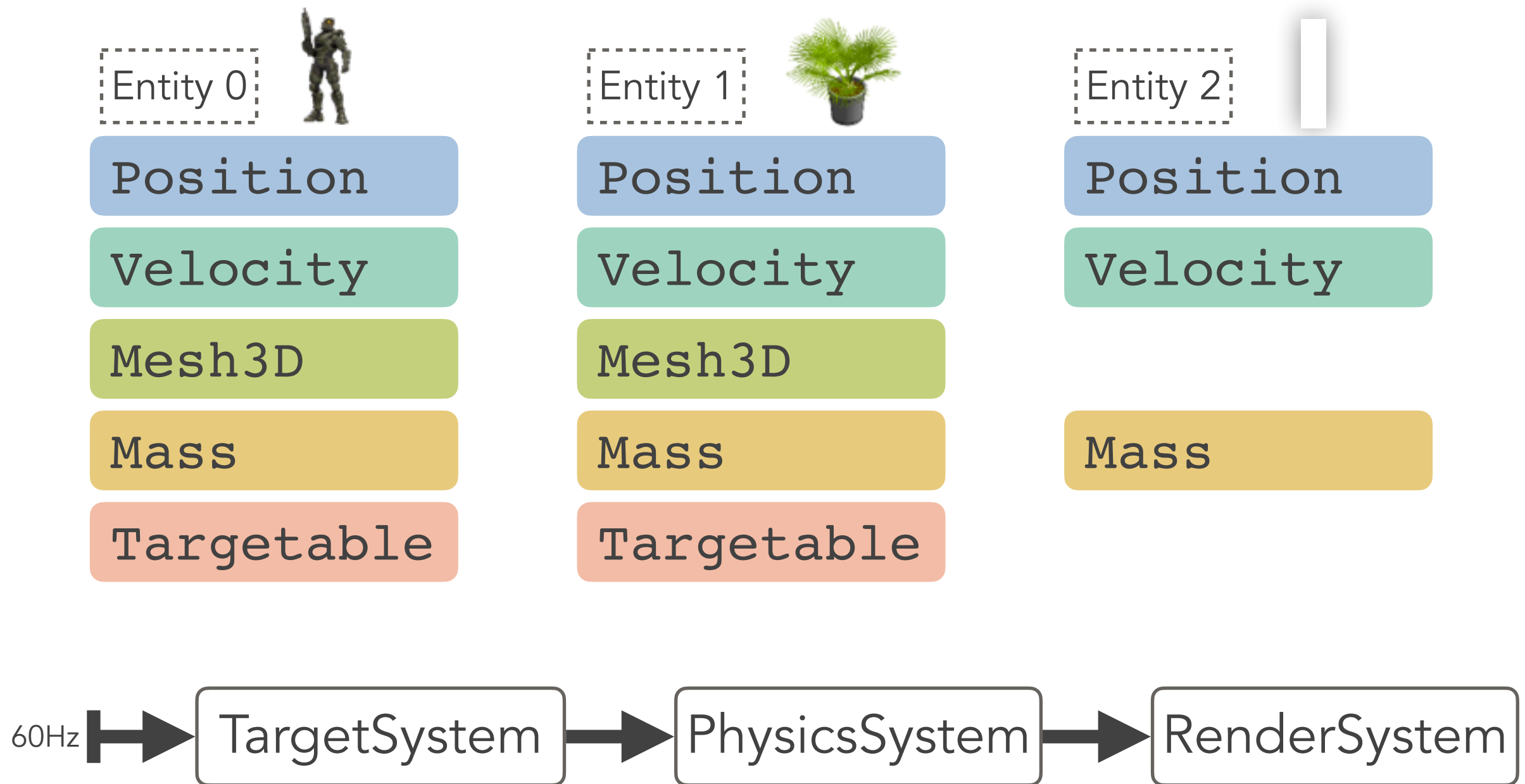
Entity 1

Entity 2

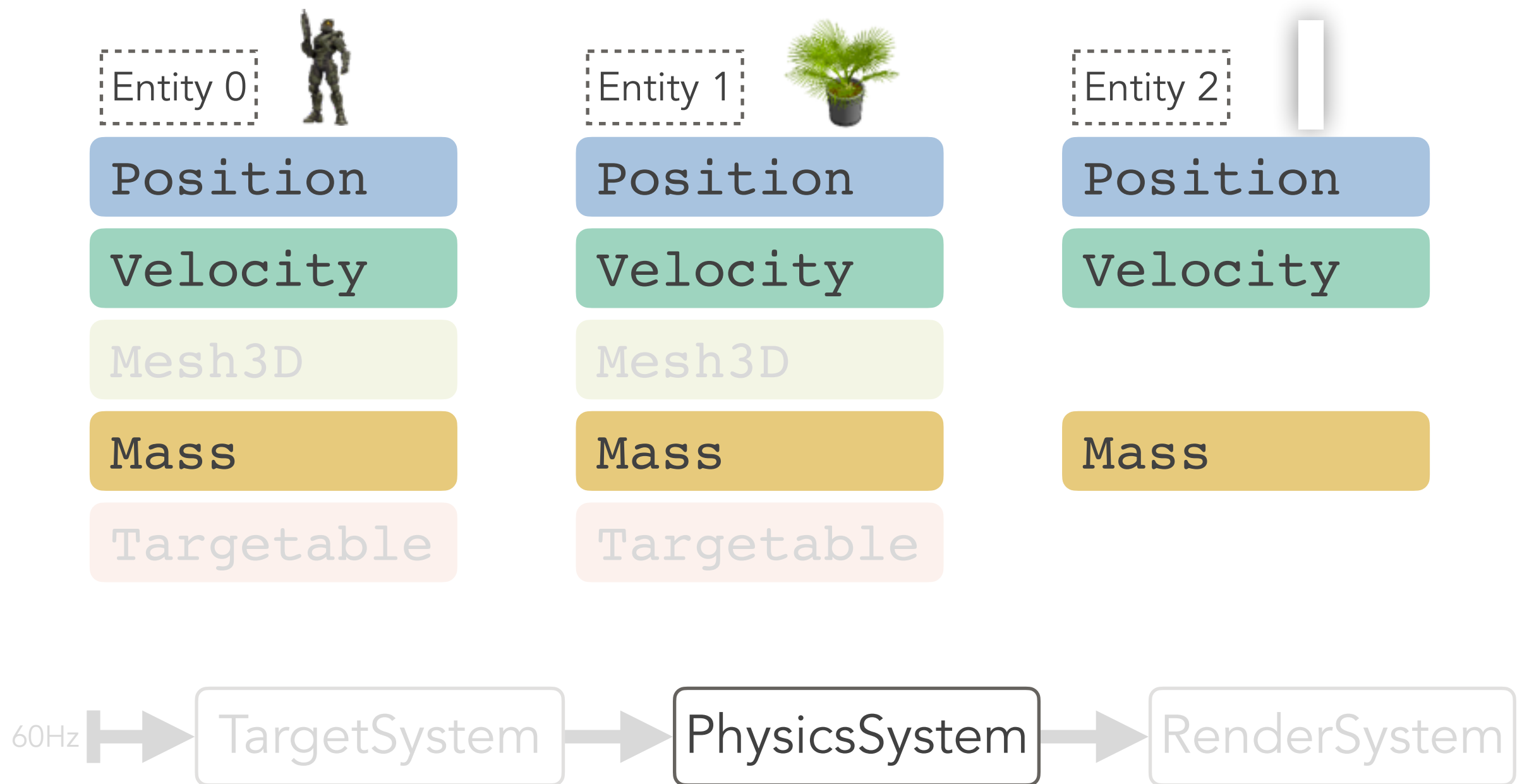
Entity-Component-System



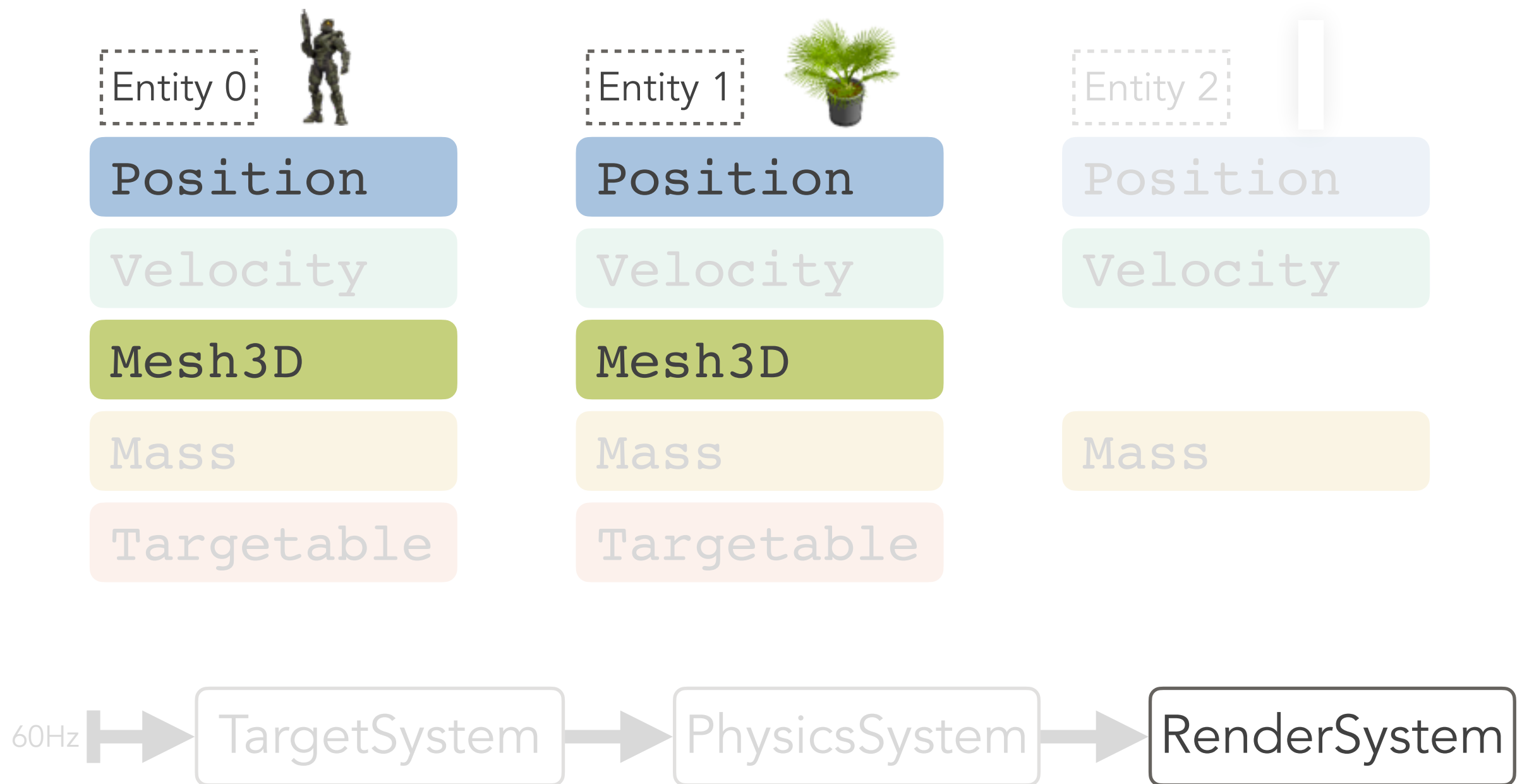
Entity-Component-System



Entity-Component-System



Entity-Component-System



Why should we care?



Thief: The Dark Project (1998)



Operation Flashpoint: Dragon Rising (2007)



Unity (2019)



Dungeon Siege (2002)



Overwatch (2016)

Why should we care?

→ ↻ ⓘ <https://forum.unity.com/threads/gui-in-pure-ecs-projects.530578/> 🌐 ★ 👤


I would guess that most of us are just doing very bare bones "detecting clicks and touches on sprites" at the moment, but has anyone come across a true UI framework for ECS yet? Or have any ideas about where you intend to go with this? Are you writing your own input fields that handle mobile keyboards and the whole nine yards? Writing everything from square 1 seems a daunting and wasteful task, but is it unavoidable?


← → ↻ ⓘ <https://love2d.org/forums/viewtopi...> ☆ 👤 ⋮

ECS for GUI


📅 Sat Oct 10, 2015 10:21 am


So guys I have been thinking about GUIs recently. I have seen tutorials for Unity's GUI system from 4.6+ and it reminds me of an ECS. In reality it is probably done with classes and not entities but considering lua is better fit for ECS rather than OOP maybe an ECS based GUI would be a good thing? One benefit of an ECS based GUI would be that the output of a hypothetical GUI editor would be pretty easy to read and make. But what are yall opinions on a ECS based GUI?

 **gamedev.net** 📄 🔍 ☰

 **Paragon123** 📈 620
Posted July 26, 2016 🔗

Does anyone have any tips, or articles about creating GUIs in an ECS system?

 **gamedev.net** 📄 🔍 ☰

 **Tangletail** 📈 2915
Posted July 26, 2016 🔗

ECS is the LAST pattern you'd want to use for GUI. GUI gets really complicated real fast. If you've never programmed one, I'd suggest taking a look at some libraries to see just how involved it can actually get.

→ ↻ ⓘ <https://forum.unity.com/threads/gui-in-pure-ecs-projects.530578/>

It is possible to make UI in ECS. Still I dont think anybody will do it anytime soon.

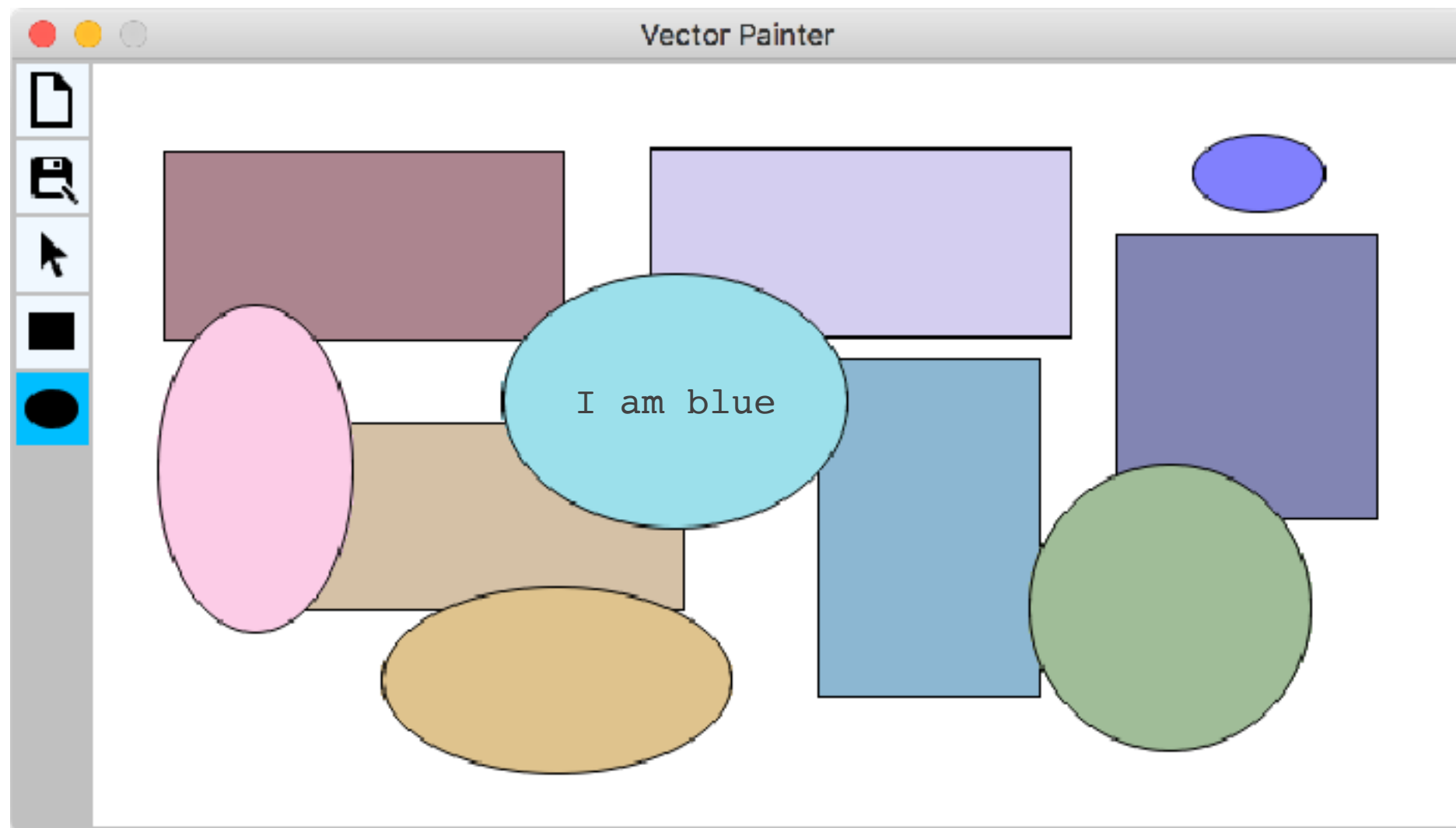
Apr 1, 2019

Programming UIs with ECS

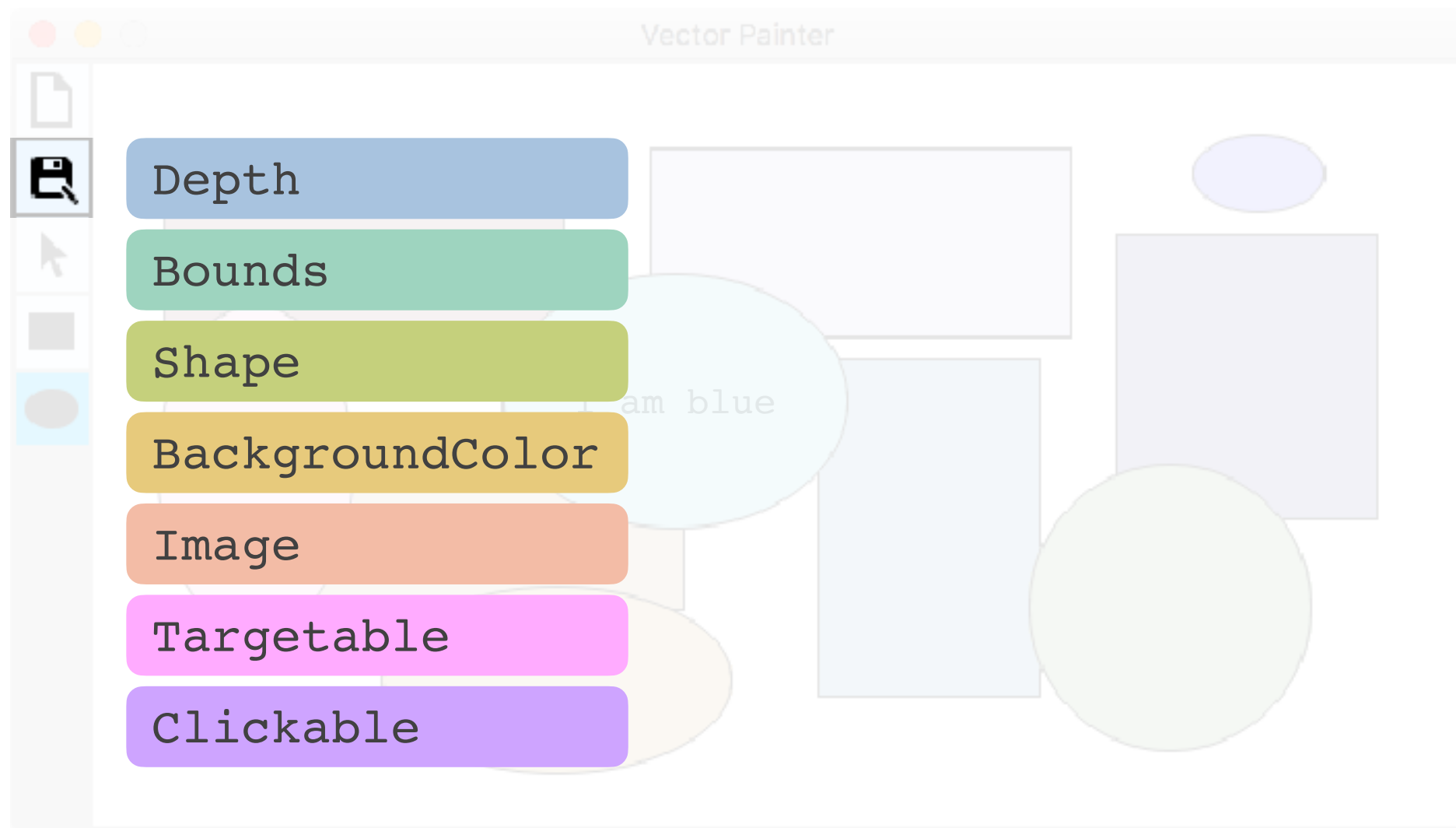
1. ECS, a composition model for video games
2. Polyphony, an experimental interaction toolkit
3. Designing UIs with *composition over inheritance*
4. Contributions, and future work

Illustrating Polyphony

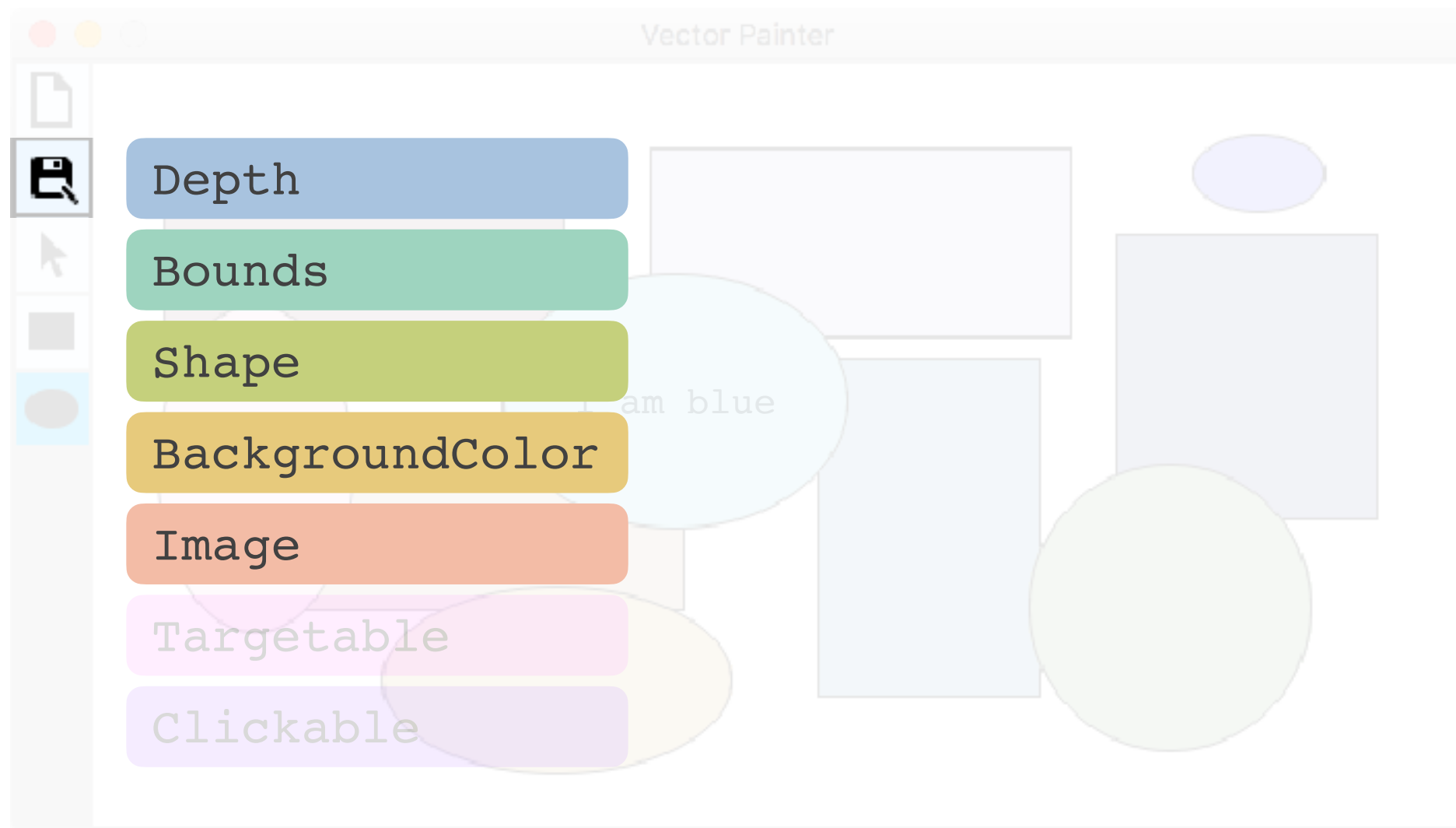
UI toolkit for Node.js, using SDL and GFX



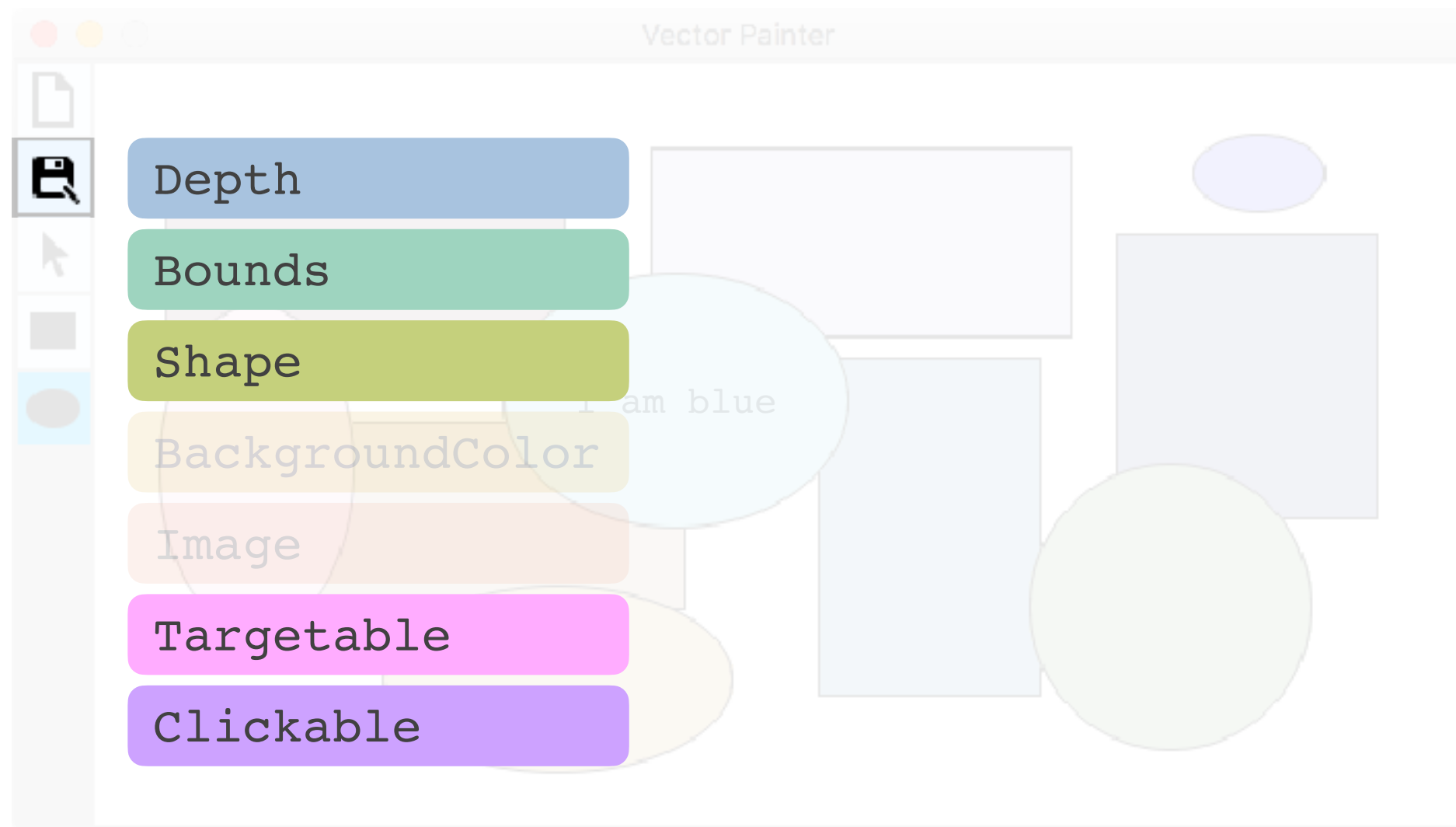
Illustrating Polyphony



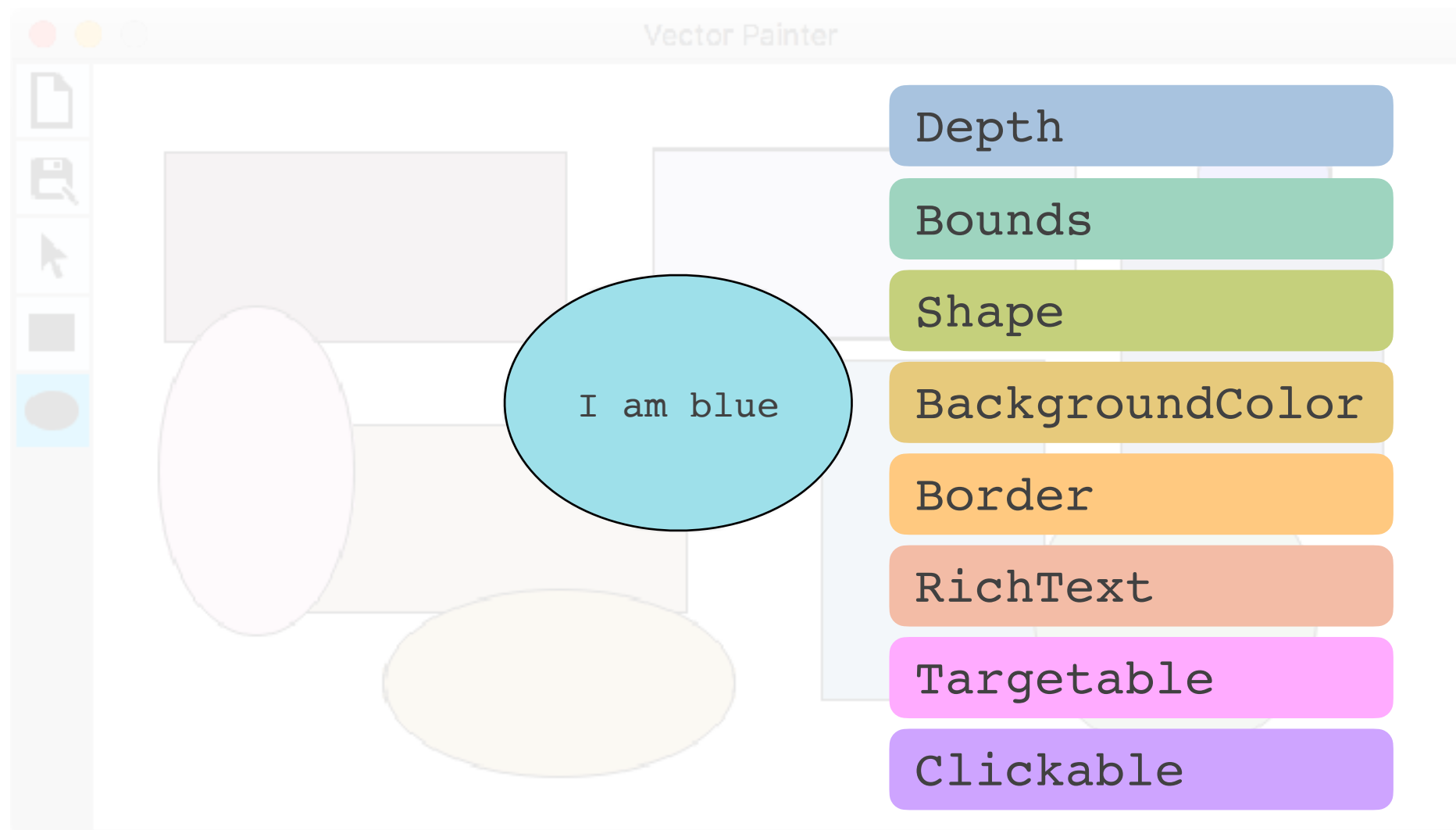
Illustrating Polyphony



Illustrating Polyphony



Illustrating Polyphony



Device Entities

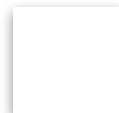


CursorPosition

Buttons



KeyStates



Bounds

Origin

The chain of Systems

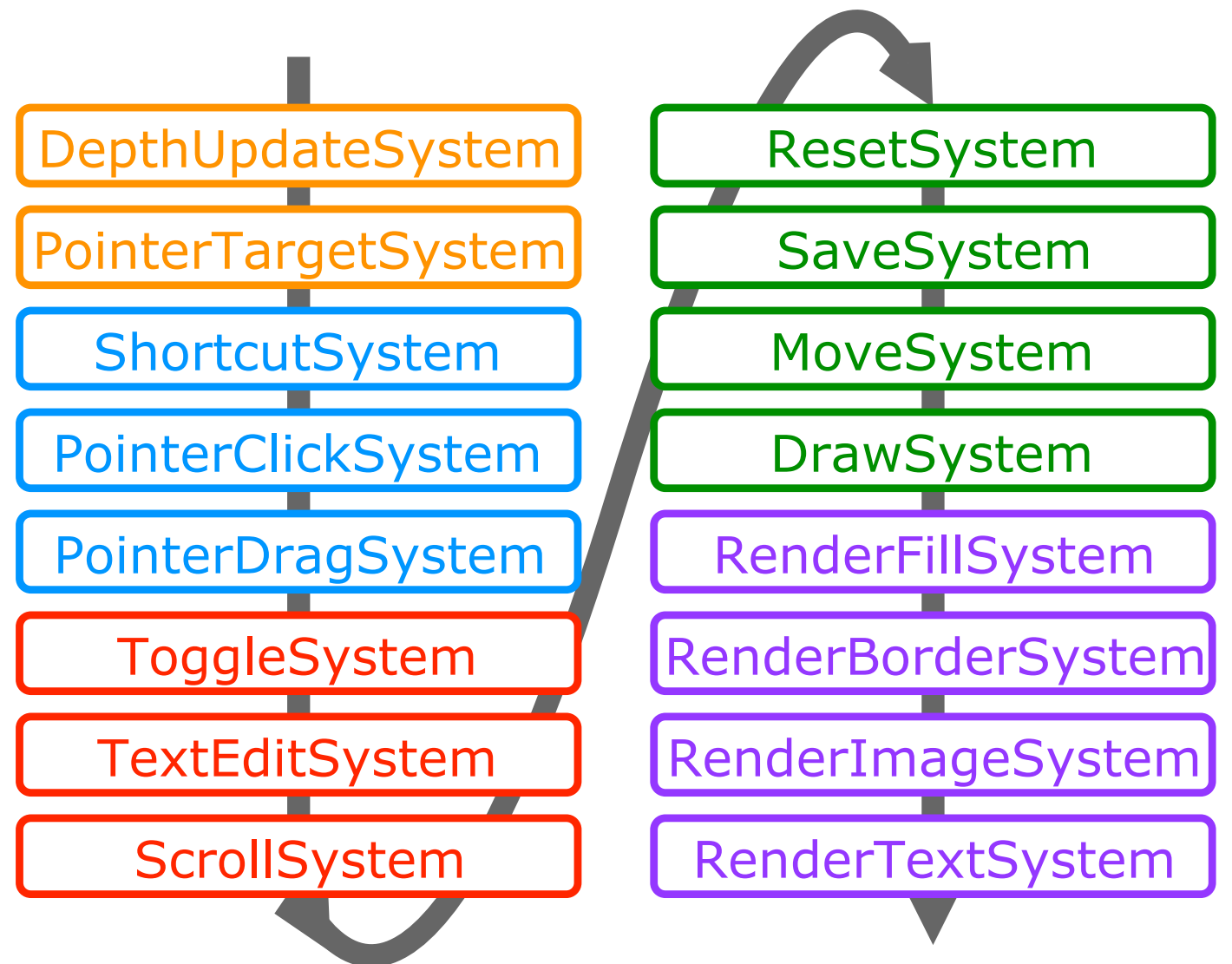
Input management

Interaction techniques

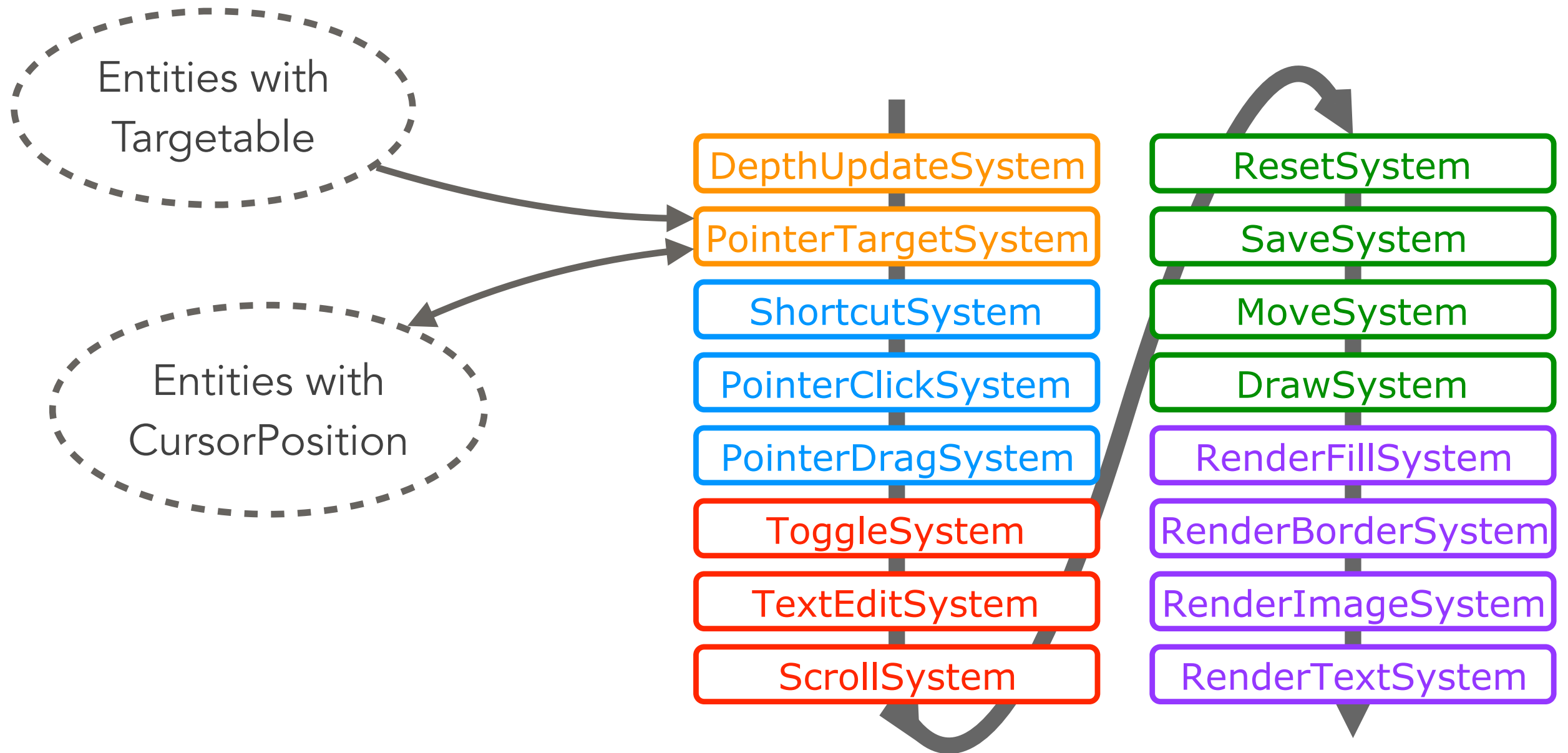
Widget-specific

Application-specific

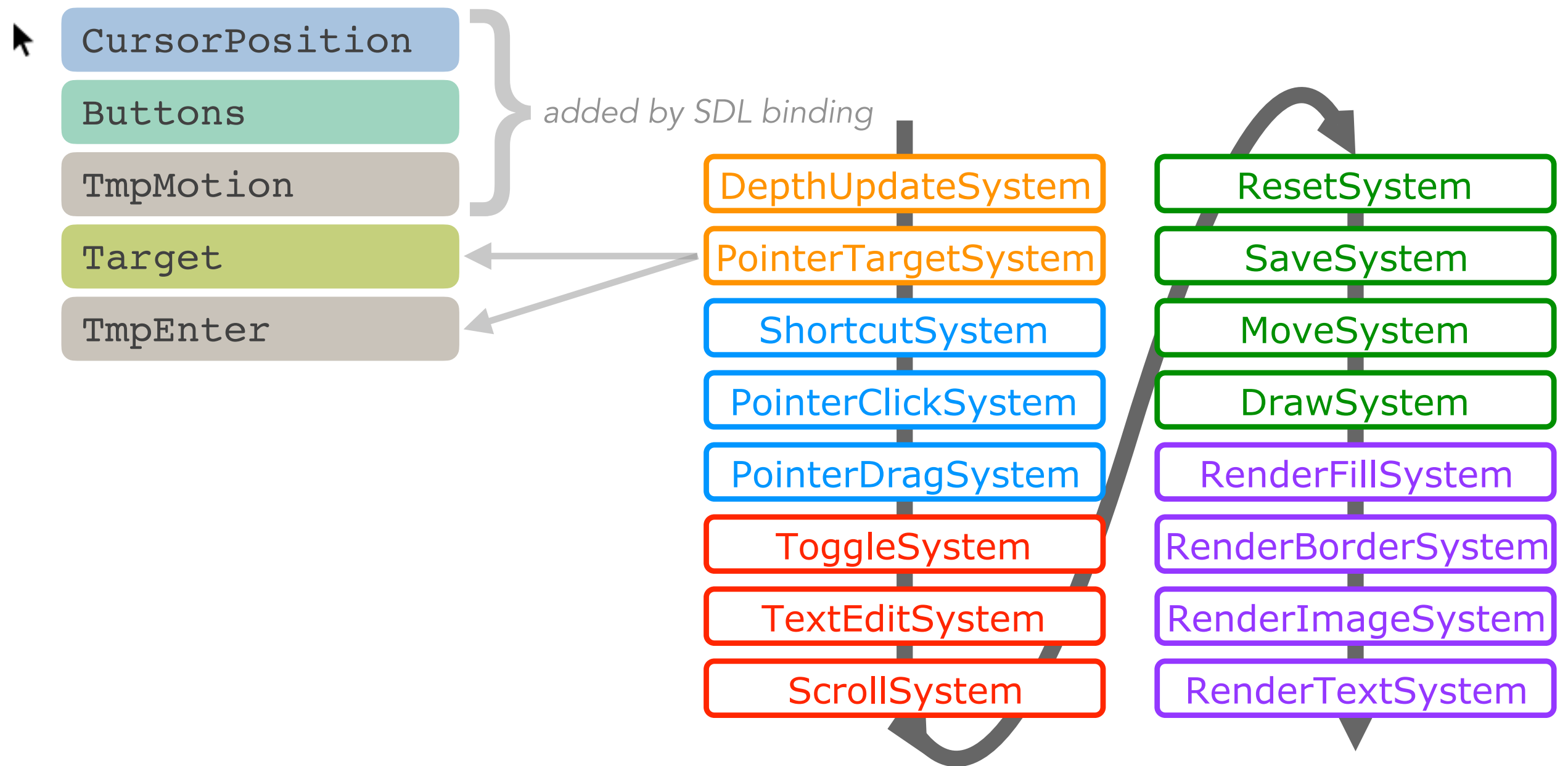
Output rendering



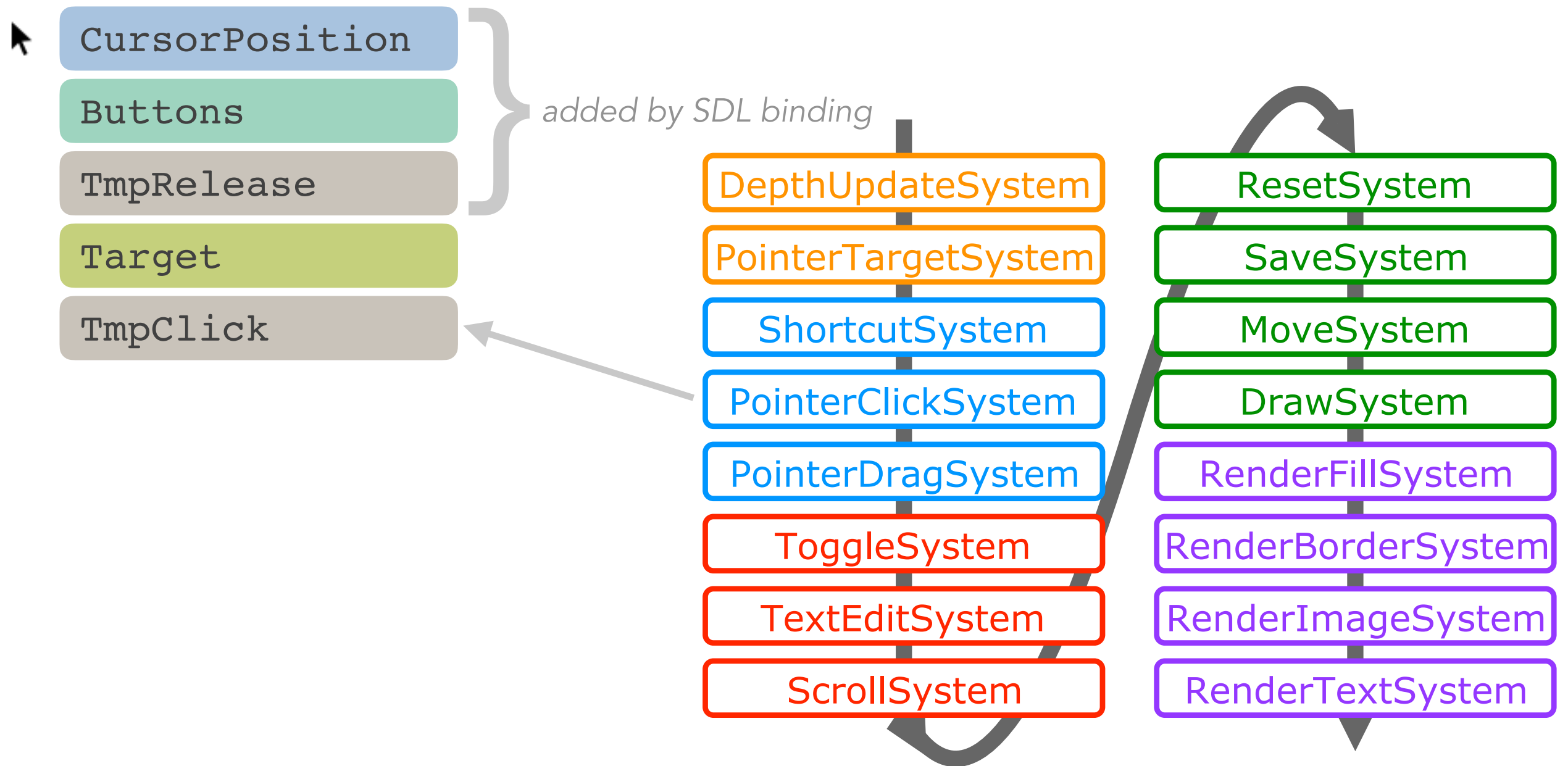
The chain of Systems



The chain of Systems



The chain of Systems



Programming UIs with ECS

1. ECS, a composition model for video games
2. Polyphony, an experimental interaction toolkit
3. Designing UIs with *composition over inheritance*
4. Contributions, and future work

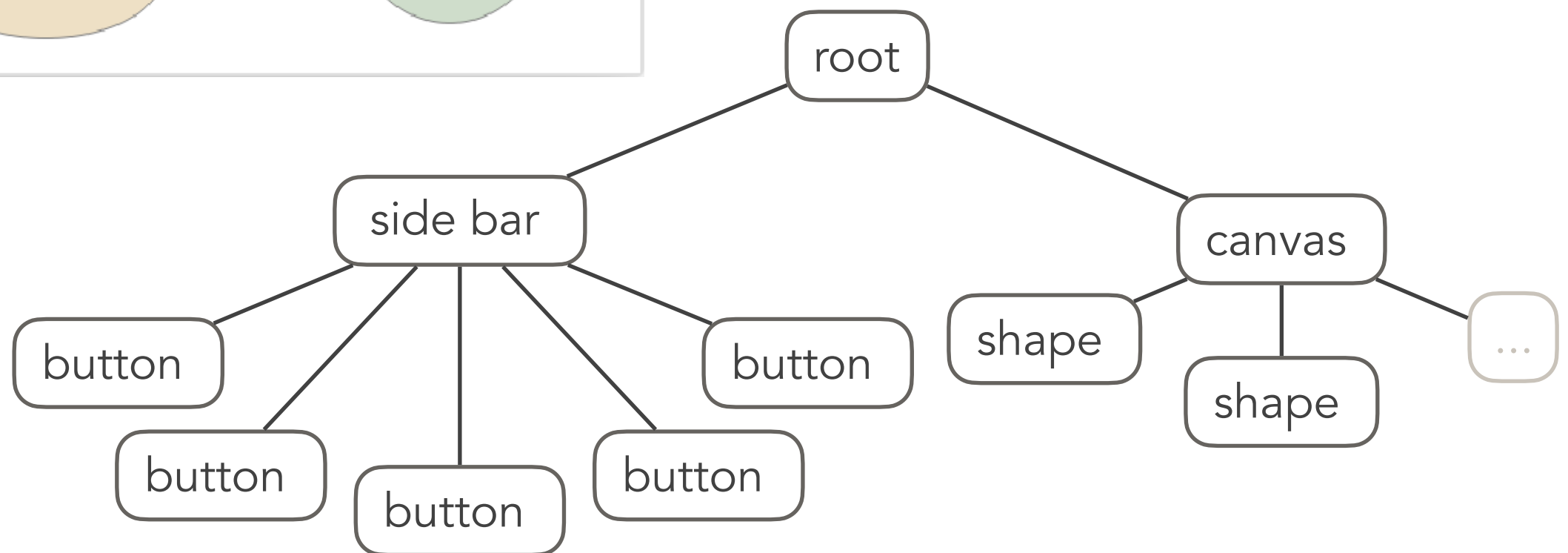
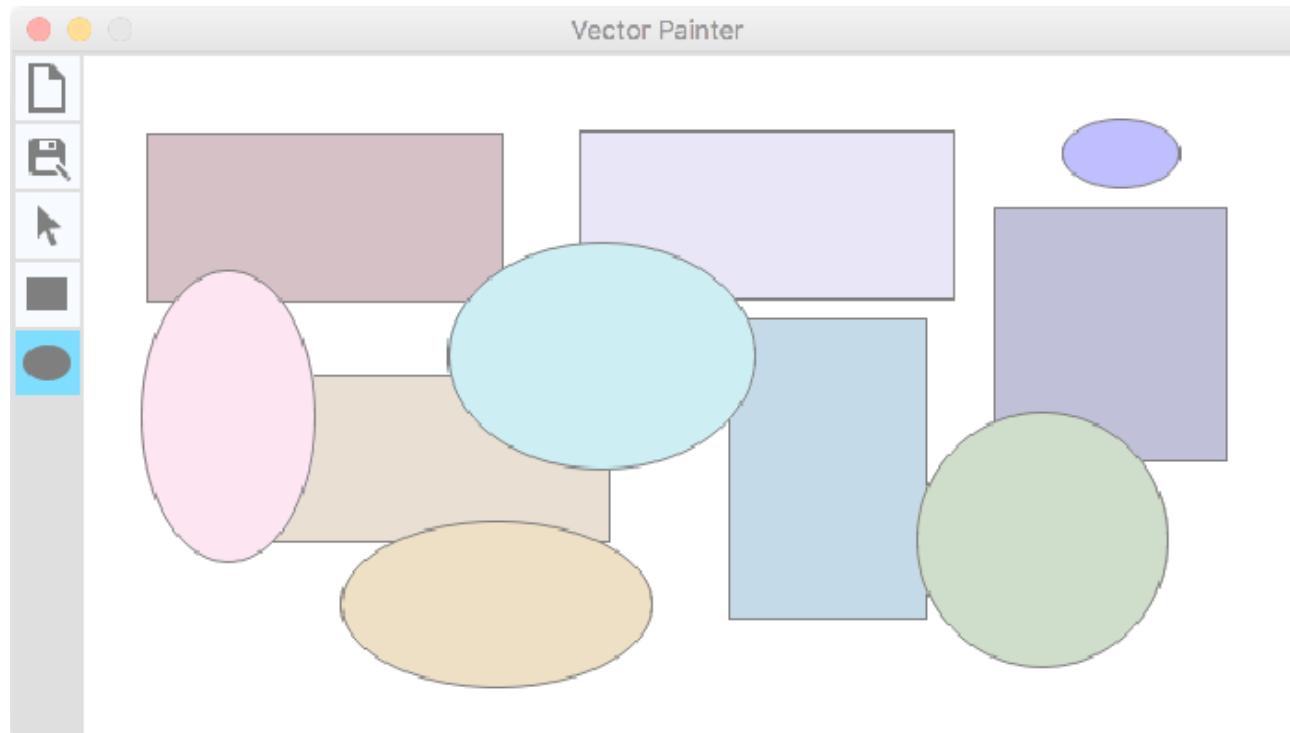
Composition over inheritance

More than one “parent” per element → *Do we need it?*

3 hierarchies:

- scene tree
- type tree
- interaction graph

The scene tree



The scene tree

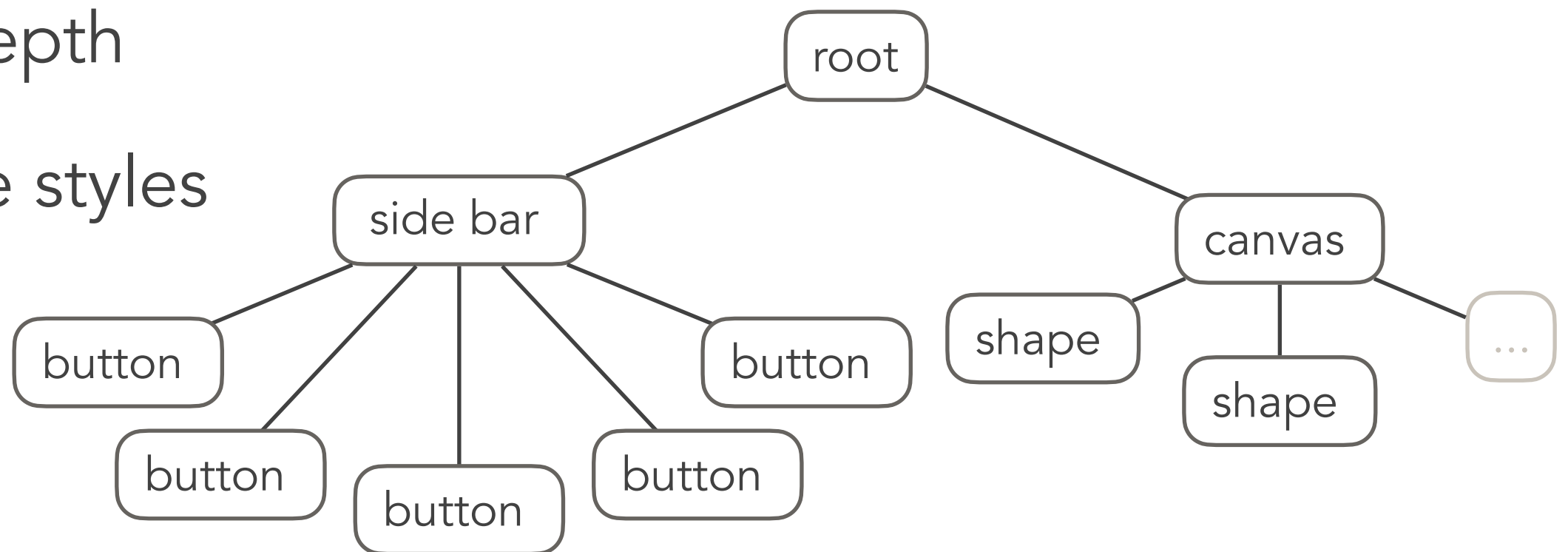
~~Iterate over nodes~~ **built-in with ECS**

Explicit references

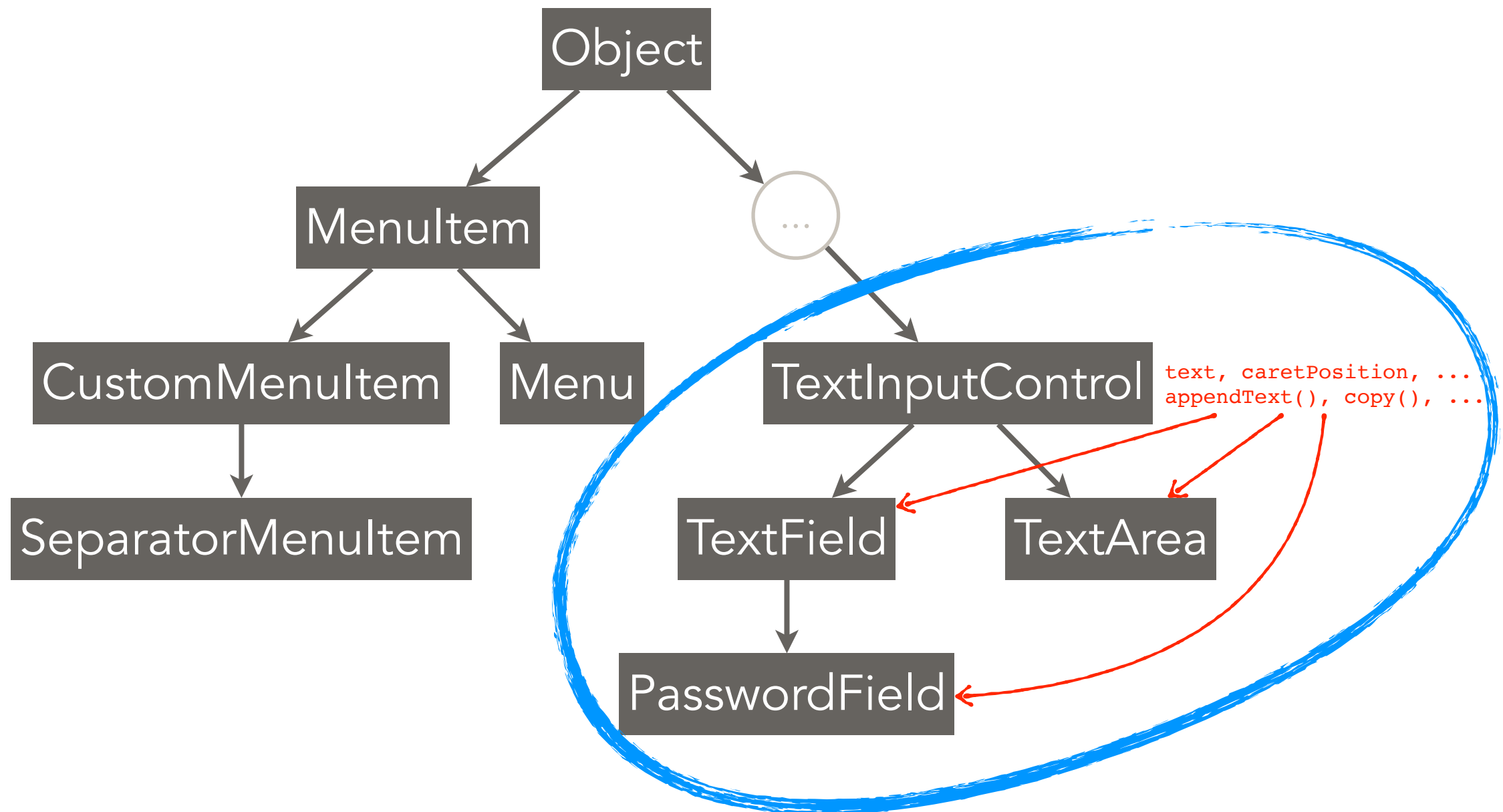
~~Relative layout~~ **global positioning (e.g. Cassowary)**

Display depth

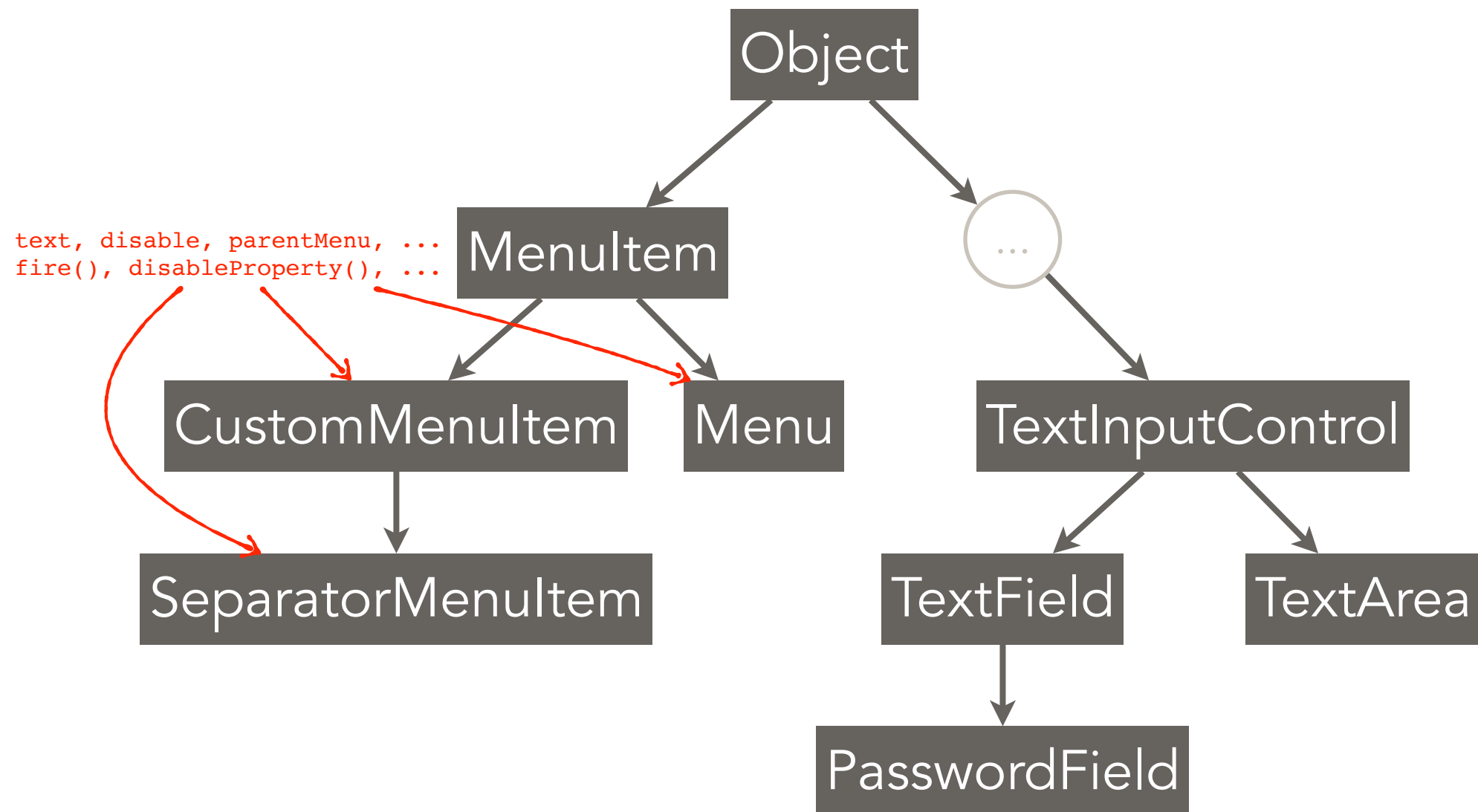
Propagate styles



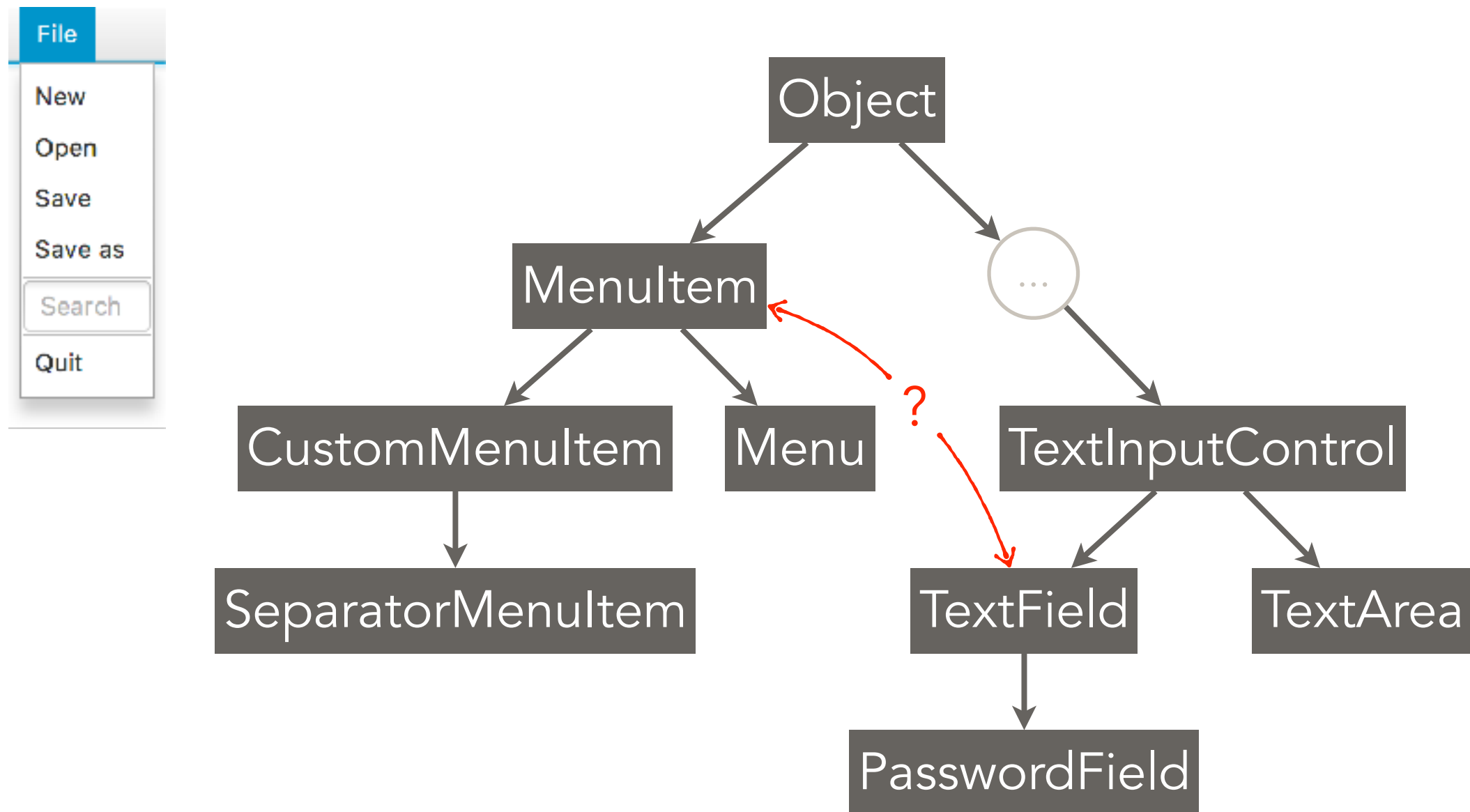
The type tree



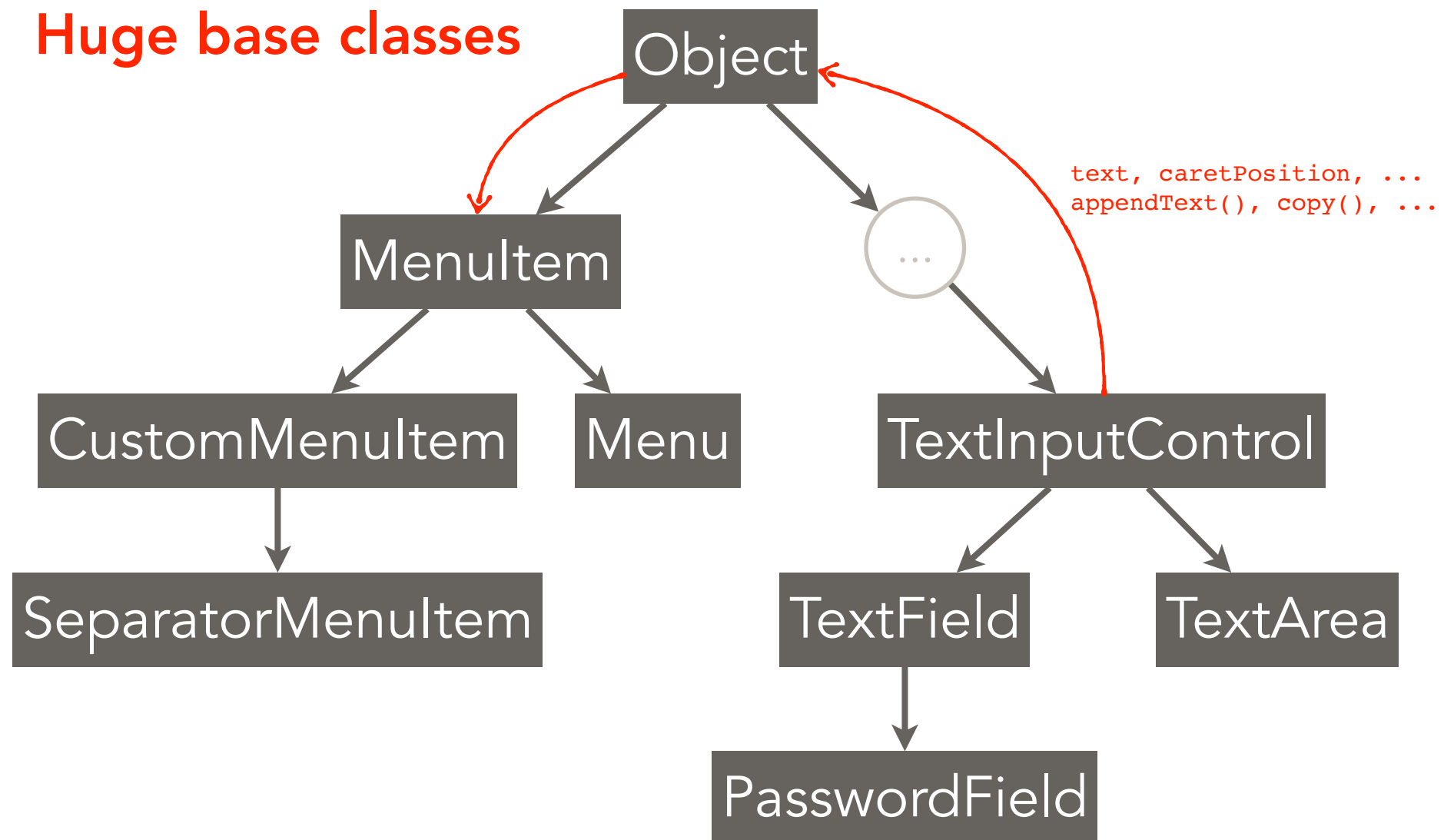
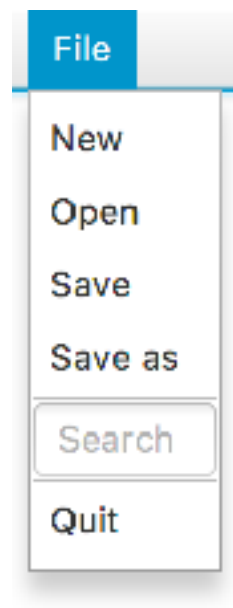
The type tree



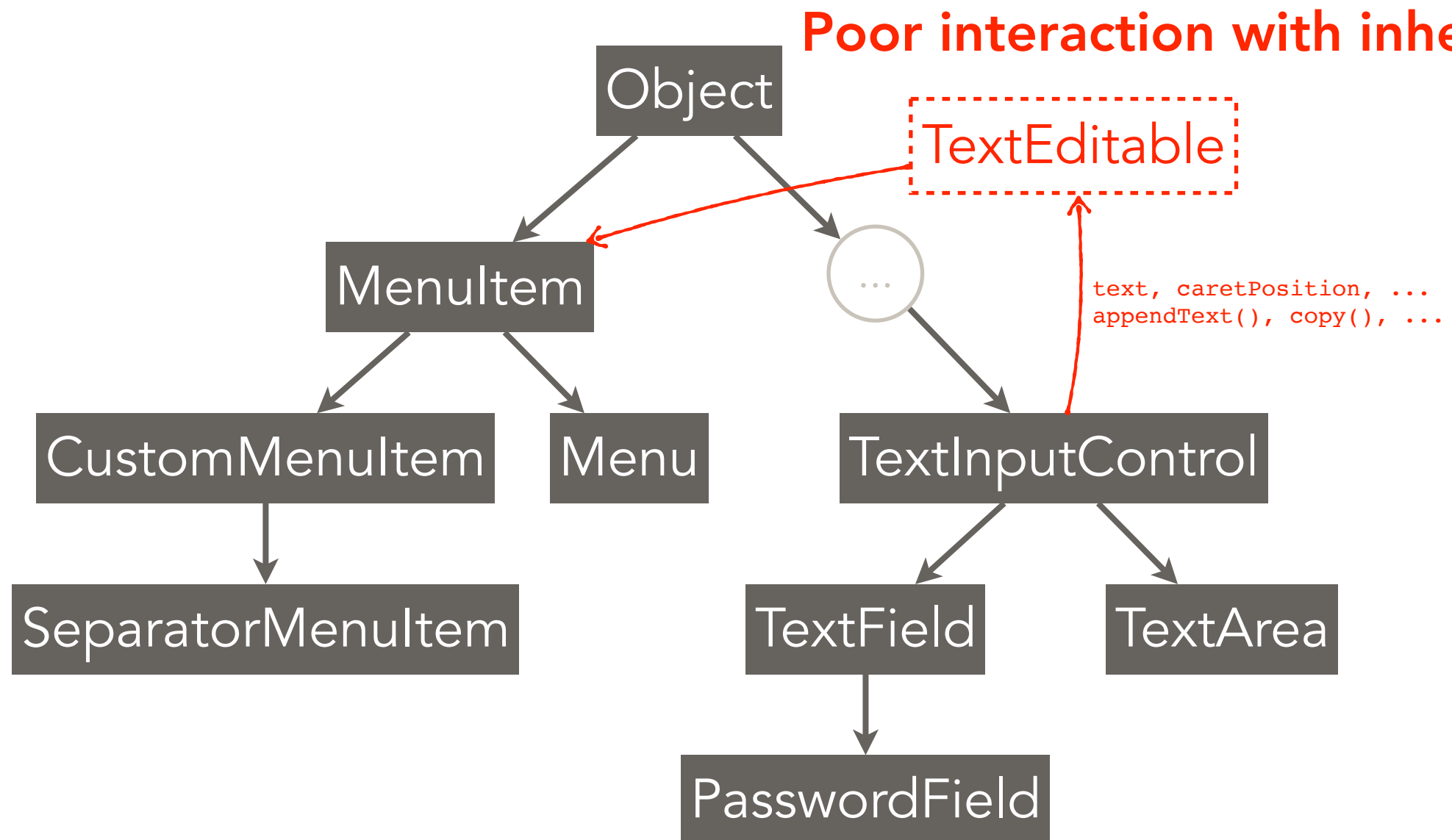
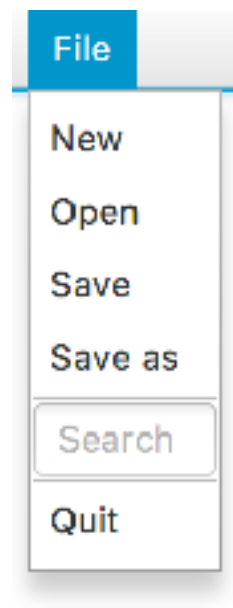
The type tree



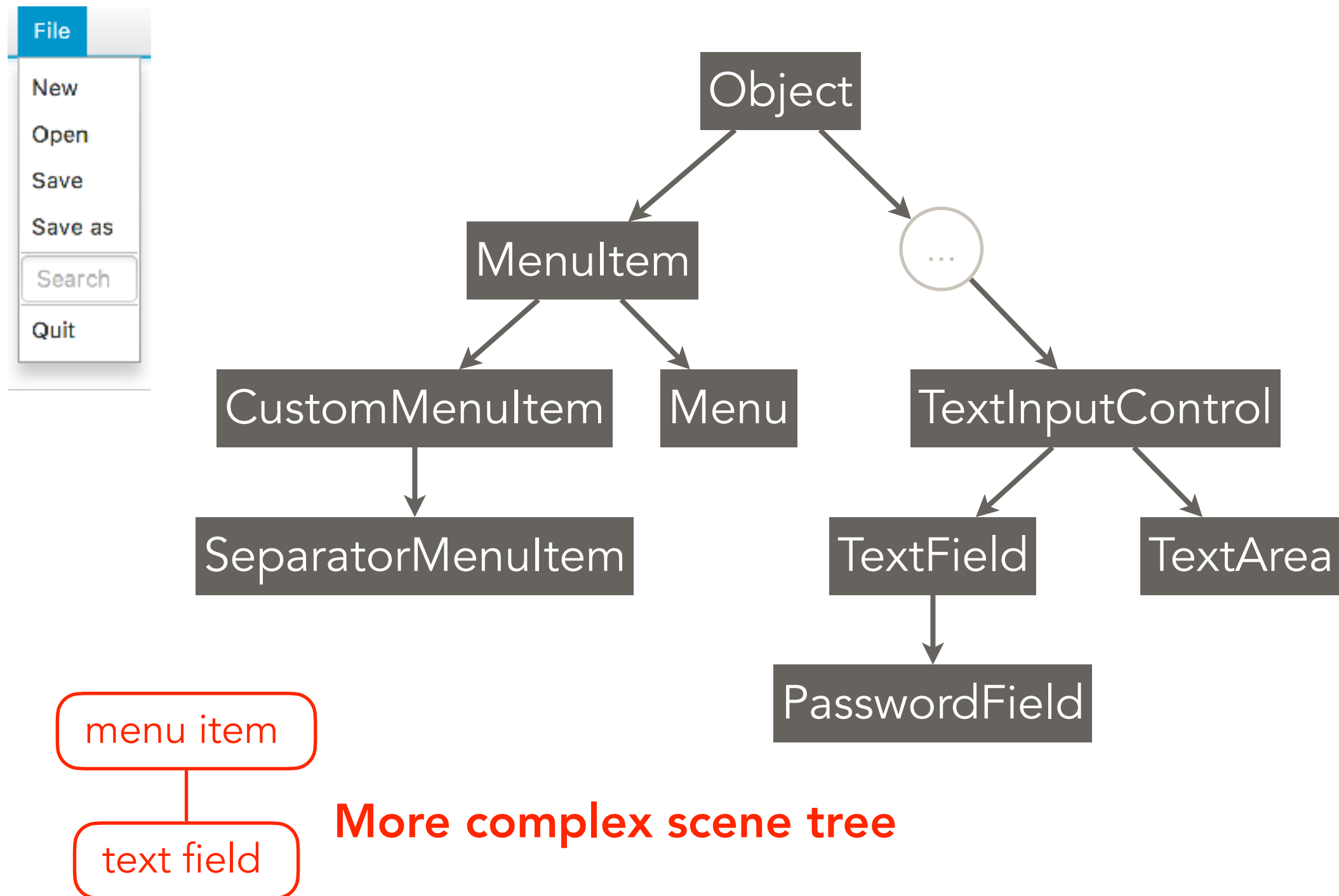
The type tree



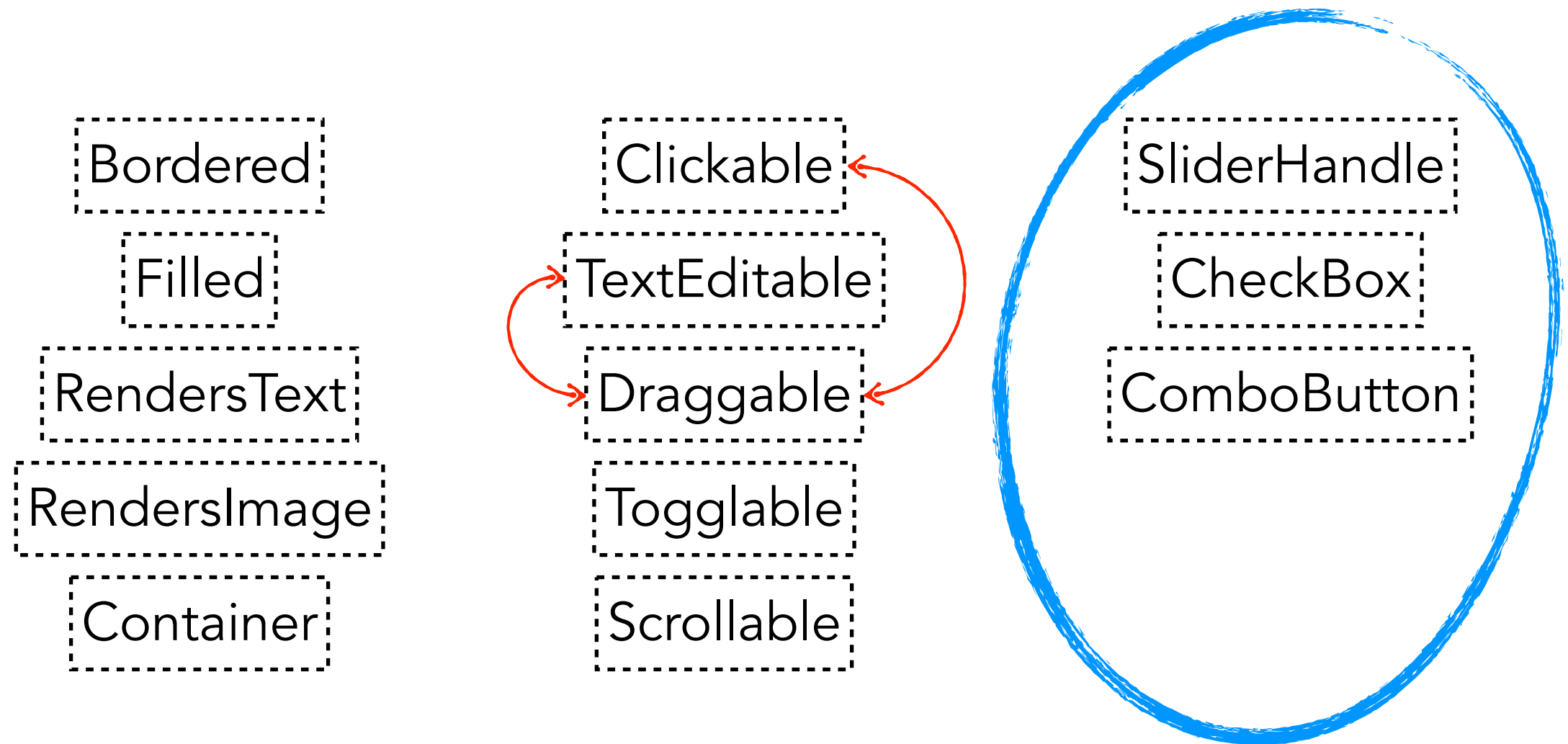
The type tree



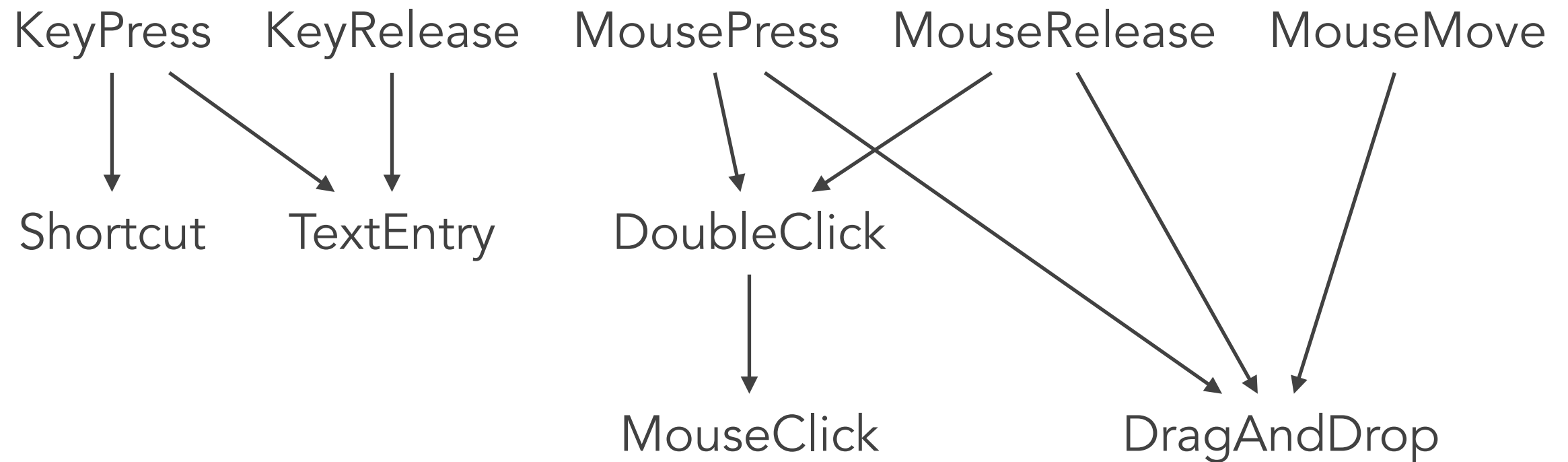
The type tree



The type tree



The interaction graph



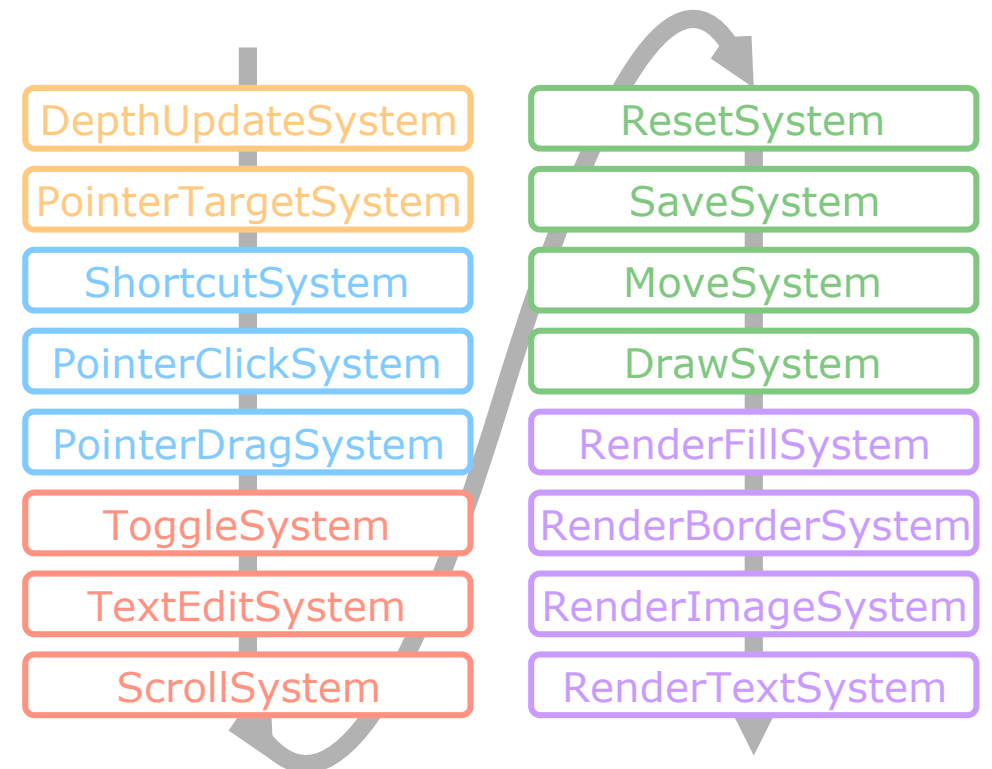
Programming UIs with ECS

1. ECS, a composition model for video games
2. Polyphony, an experimental interaction toolkit
3. Designing UIs with *composition over inheritance*
4. Contributions, and future work

No callbacks

All of the logic is contained inside Systems

Systems react to events through *temporary* Components



👍 No “spaghetti of callbacks” (Myers, 1991)

👎 Less control over the code of widgets (appearance)

🤔 Bigger functions

No event structures

Event occurrence is signalled on device
Entities

Event propagation with permanent and
temporary Components



CursorPosition

Buttons

TmpReleased

Target

TmpClick

- 👍 No possibility of queuing/delaying events
- 👍 Preserve Components of specific devices (pressure touch)
- 👎 More work to pass events across applications/machines

Adapting ECS to UIs

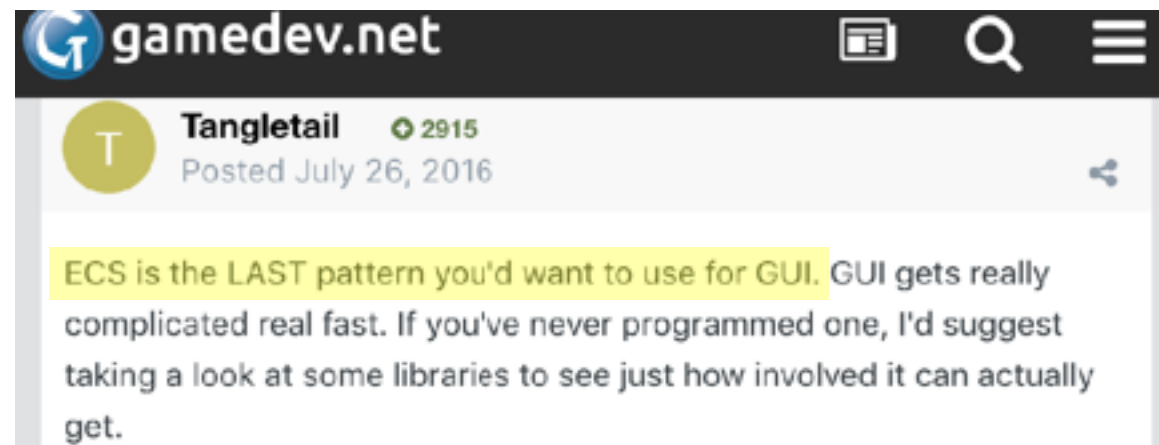
Systems are Entities (modelling dependencies with Components)

Systems chain filtering (support for multimodality)

Device Entities (supporting multiple devices)

Temporary Components (reacting to events without callbacks)

Future works



Use article to promote support for interaction in ECS

Build more complex UIs

Contribute to future ECS language

Links

<https://gitlab.inria.fr/Loki/PolyphonyECS/>

https://www.gamasutra.com/view/feature/131762/postmortem_thief_the_dark_project.php

<http://t-machine.org/index.php/category/entity-systems/>

<http://entity-systems.wikidot.com/>

<https://unity3d.com/learn/tutorials/topics/scripting/introduction-ecs>

<http://bit.ly/2Zzl6rc> ← programming interaction study

OOP vs ECS

OOP	ECS
an object stores its field contiguously in memory	Components may be stored contiguously by Entity, or by type
execution stream is a series of messages between objects	execution stream is a sequence of procedures
objects are "visible" by keeping a reference to them	Entities are "visible" by requesting a set of Components
an object's "nature" is determined by the <i>types</i> of inherited classes	an Entity's "nature" is determined by the Components bound at any time
implicit deletion (lexical scope, garbage collection), or explicit (C++)	explicit deletion

ECS in video games

Independence of artists, designers, and programmers
(Leonard, 1999)

Data-oriented design (Acton, 2014)

→ abstract data storage to optimise memory access

Parallel processing on multicore systems (Unity, 2018)

Limits

Very sensitive to choice of Components

Less flexible than object-based toolkits

Difficult to implement (unadapted languages, unclear descriptions)

Components

Components	Entity Factories						
	Button	Canvas	Shape	Pointer	Keyboard	View	Systems
children		×					
depth	×	×	×				
bounds	×	×	×			×	
shape	×	×	×				
backgroundColor	×	×	×				
border			×				
image	~						
richText	~		~				
targetable	×	×	×				
clickable	×						
toggleGroup	~						
draggable			×				
textEditable			×				
cursorPosition				×			
buttons				×			
keyStates					×		
focus					×		
origin						×	
viewport						~	
scrollable						~	
runOn							×
order							×

Things covered in the paper

Syntax of Polyphony

Implementation of drag&drop technique

Implementation choices against 3 related works

Pros/cons of ECS for UI programming

Managing *behaviors* with ECS