Use.GPU

An Incremental Effect System
For Declarative/Reactive 3D Graphics

Steven Wittens

Who am I?

Who am I?

Who am I?

That guy with that website with that header

Graphics?

Graphics?

- Who has the prettiest pixels?

Graphics?

- Who has the prettiest pixels?
- Which renderer runs fastest??

Graphics?

- Who has the prettiest pixels?
- Which renderer runs fastest??
- How many triangles???

Graphics?

- Who has the prettiest pixels?
- Which renderer runs fastest??
- How many triangles???

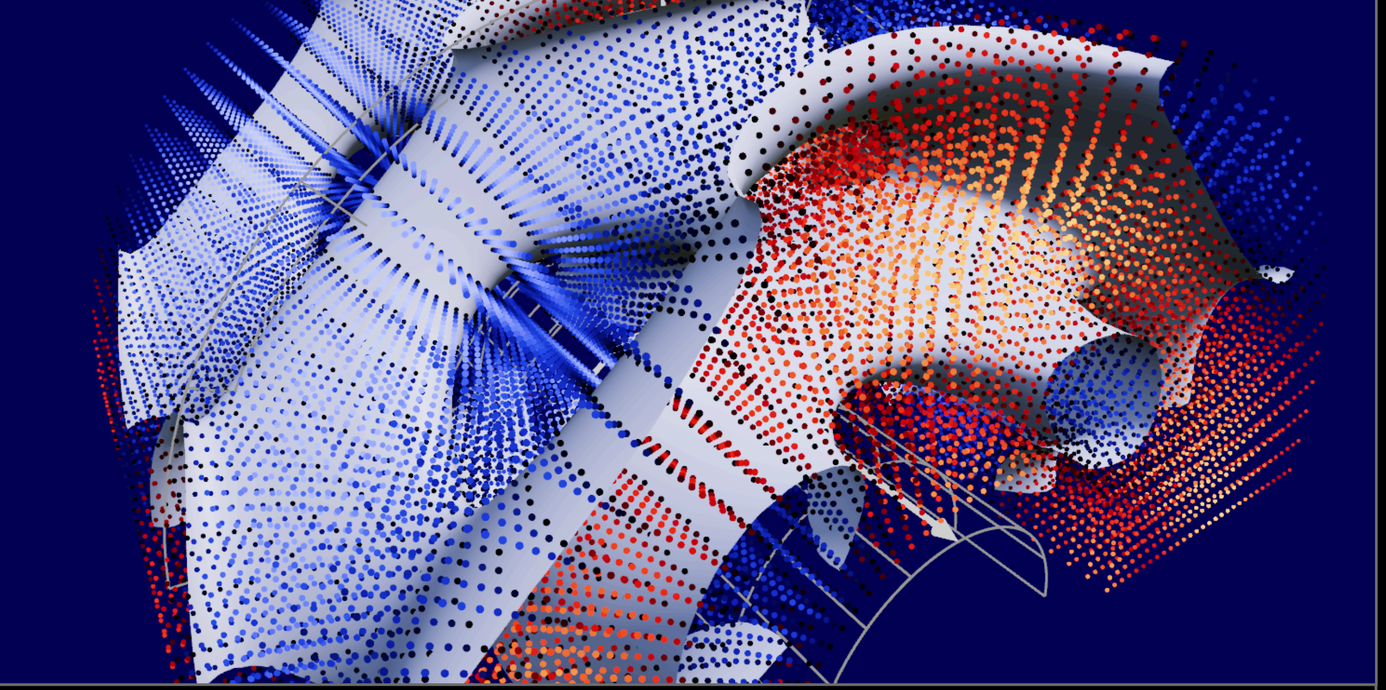
That's not what this talk is about

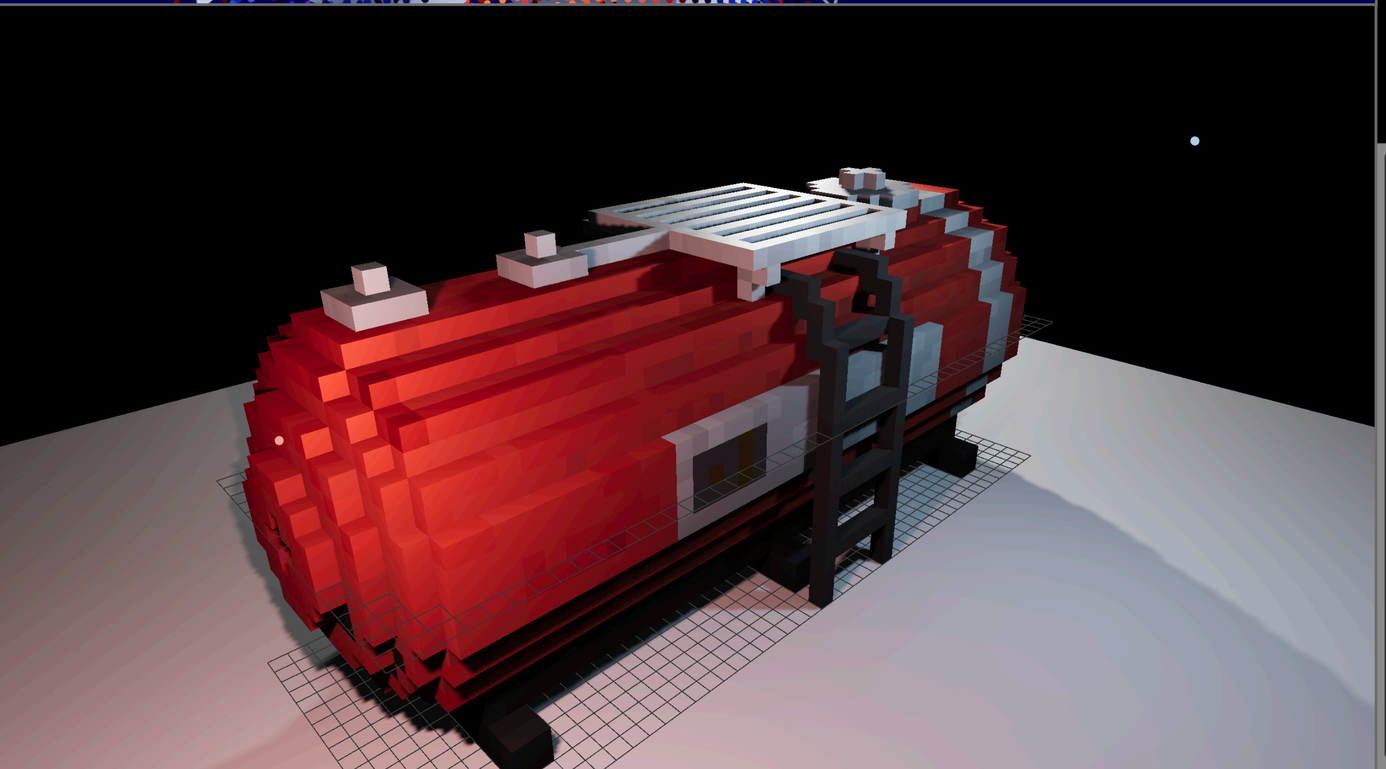
Graphics?

- Who has the prettiest pixels?
- Which renderer runs fastest??
- How many triangles???

That's not what this talk is about

Use.GPU

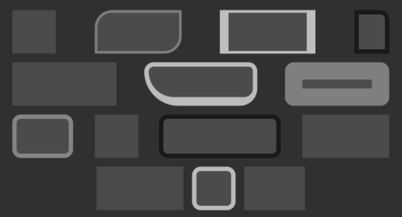




Use.GPU

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.



Lorem ipsum ⭐✨🍕

Dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

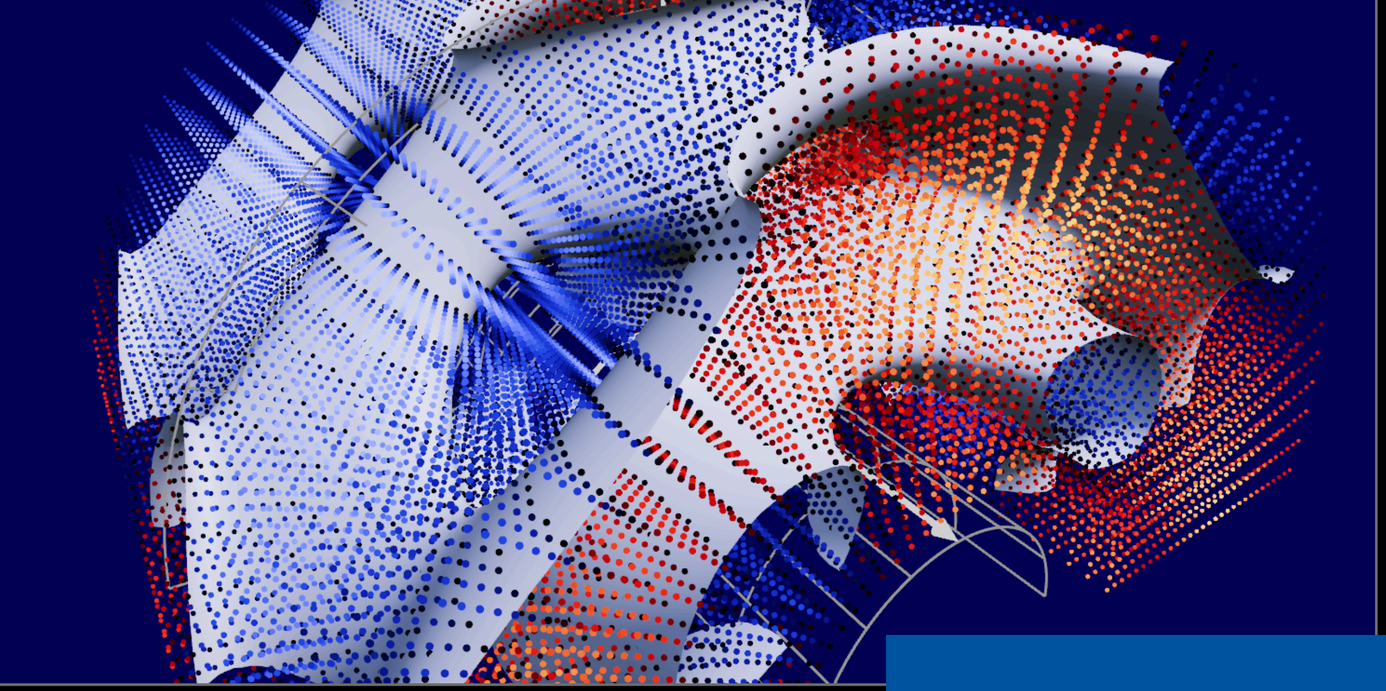
Fugiat nulla pariatur

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

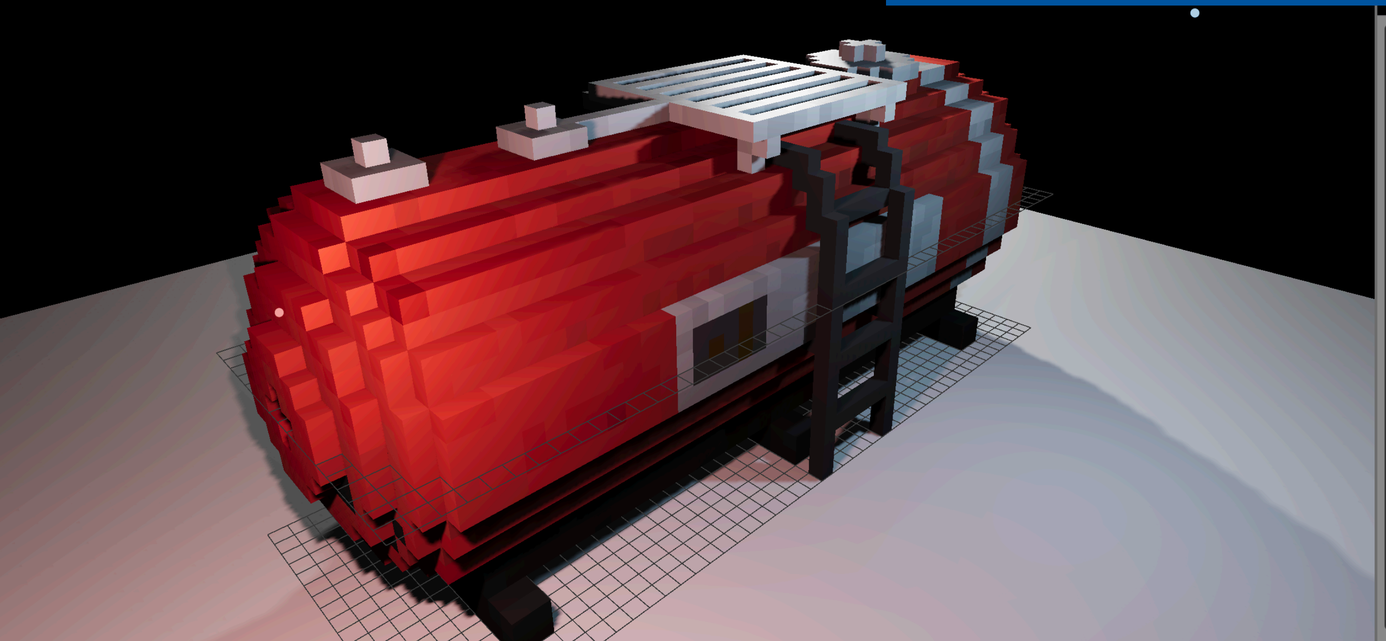
Ut enim ad minim veniam, quis nostrud Aute Cupiditat Aliquip exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.



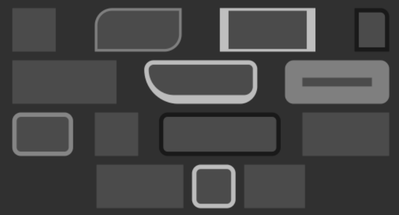
The most **versatile** pixels?



Use.GPU

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.



Lorem ipsum ⭐✨🍕

Dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Fugiat nulla pariatur

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud Aute Cupiditat Aliquip exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Client: "We want a ..."

** not affiliated*

Samara

Want to know more? — Talk to one of our experts

Configure Backyard

Make it yours. Choose your layout and finishes. Customize your windows, doors, and decks.

Backyard is currently available in California.

Choose your layout

Studio
420 gross sq. ft.
– $60,000

One bedroom
540 gross sq. ft.
– $40,000

Two bedroom
690 gross sq. ft.

$346,500
or $2,075/mo
Includes installation estimate

Confirm availability →

** not affiliated*

Samara

Want to know more? — Talk to one of our experts

Configure Backyard

Make it yours. Choose your layout and finishes. Customize your windows, doors, and decks.

Backyard is currently available in California.

Choose your layout

Studio
420 gross sq. ft.
– $60,000

One bedroom
540 gross sq. ft.
– $40,000

Two bedroom
690 gross sq. ft.

$346,500
or $2,075/mo
Includes installation estimate

Confirm availability →

** not affiliated*

Samara

The permutation problem

Want to know more? — Talk to one of our experts

Configure Backyard

Make it yours. Choose your layout and finishes. Customize your windows, doors, and decks.

Backyard is currently available in California.

Choose your layout

Studio
420 gross sq. ft.
– $60,000

One bedroom
540 gross sq. ft.
– $40,000

Two bedroom
690 gross sq. ft.

$346,500
or $2,075/mo
Includes installation estimate

Confirm availability →

** not affiliated*

Samara

The permutation problem

Every combination hand-baked and coded

Configure Backyard

Make it yours. Choose your layout and finishes. Customize your windows, doors, and decks.

Backyard is currently available in California.

Choose your layout

Studio
420 gross sq. ft.
– $60,000

One bedroom
540 gross sq. ft.
– $40,000

Two bedroom
690 gross sq. ft.

$346,500
or $2,075/mo
Includes installation estimate

Confirm availability →

** not affiliated*

Samara

The permutation problem

Every combination hand-baked and coded

Light is 'painted on'

Configure Backyard

Make it yours. Choose your layout and finishes. Customize your windows, doors, and decks.

Backyard is currently available in California.

Choose your layout

Studio

One bedroom

ss sq. ft.

00

Two bedroom

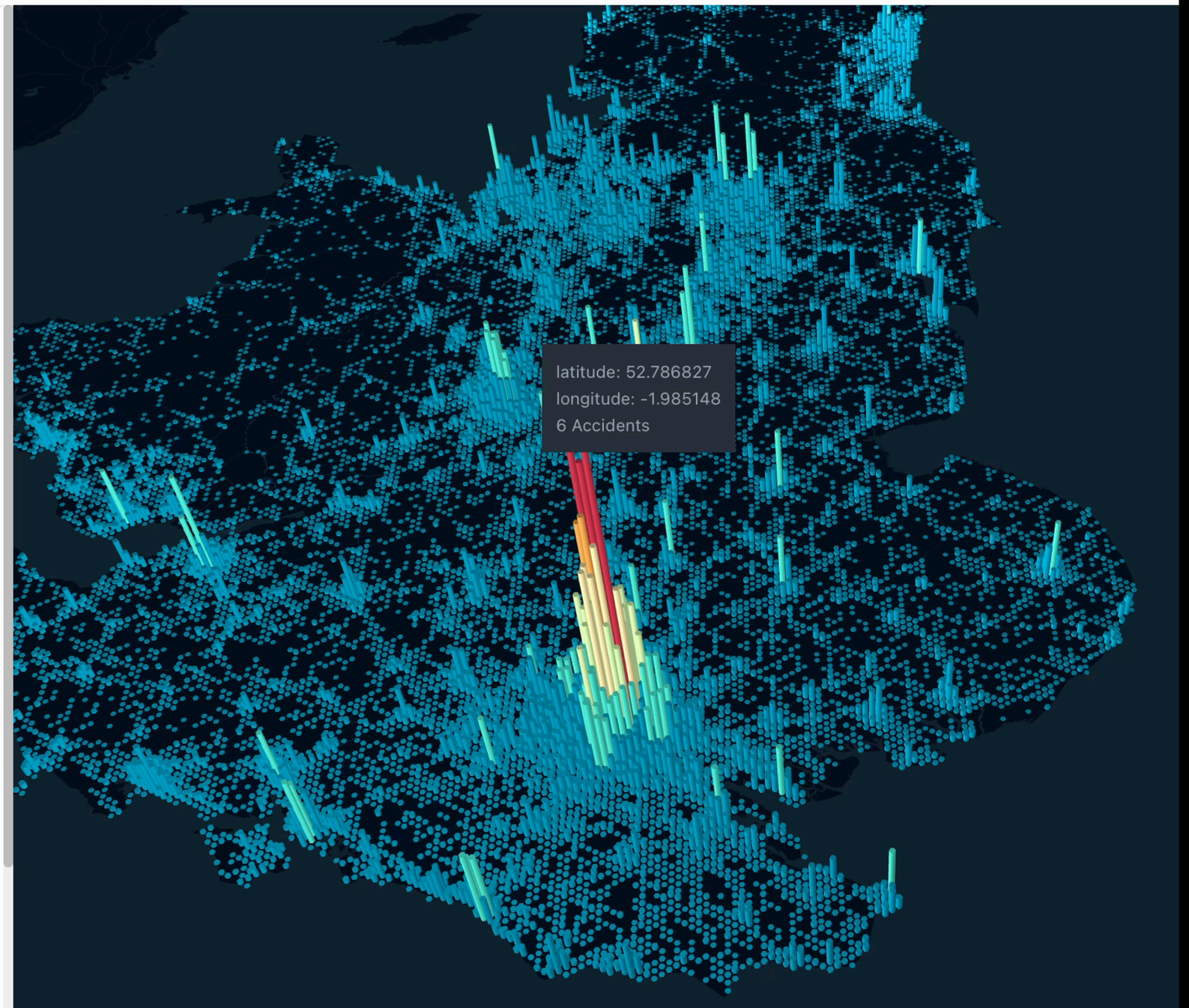
690 gross sq. ft.

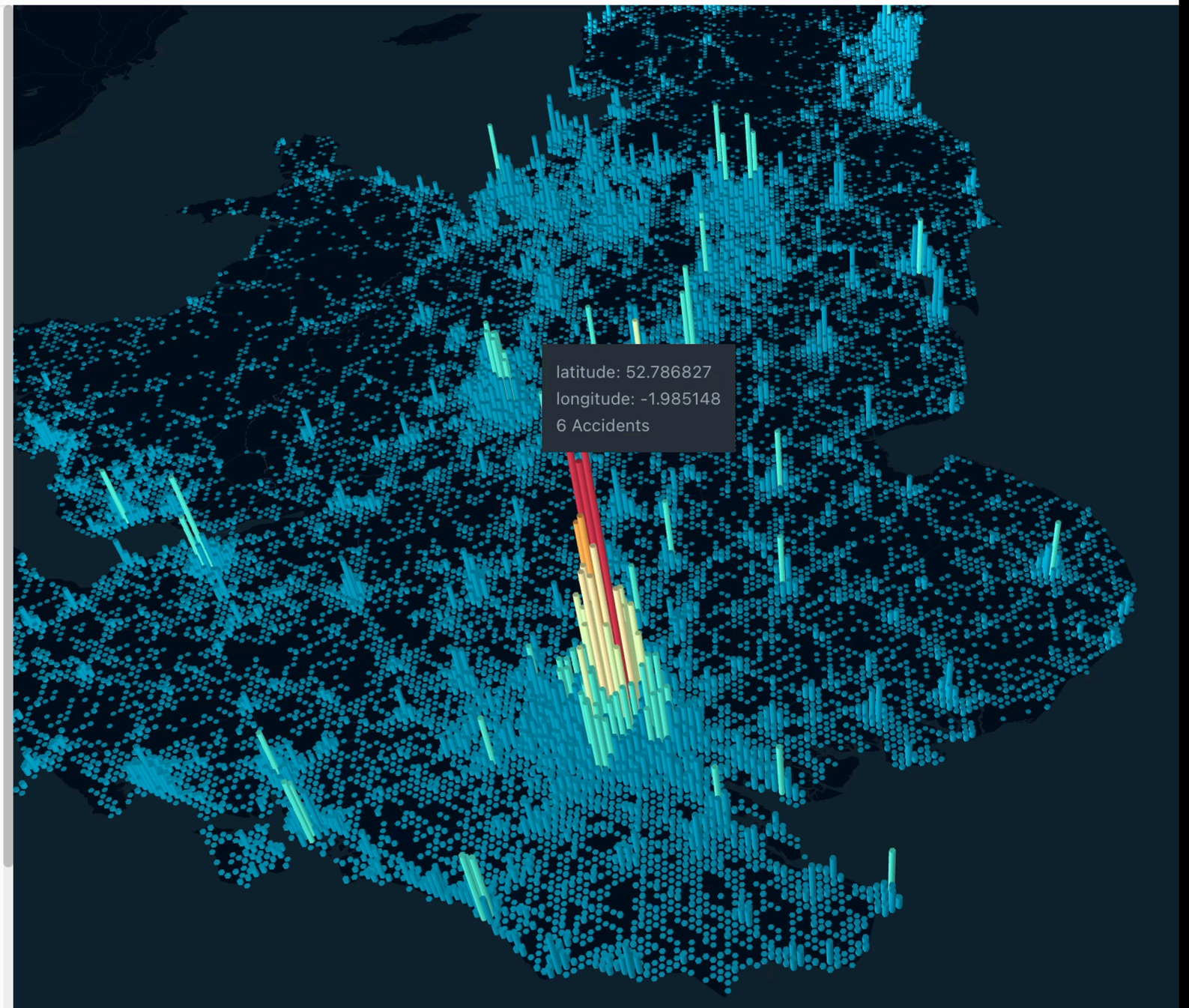
$346,500

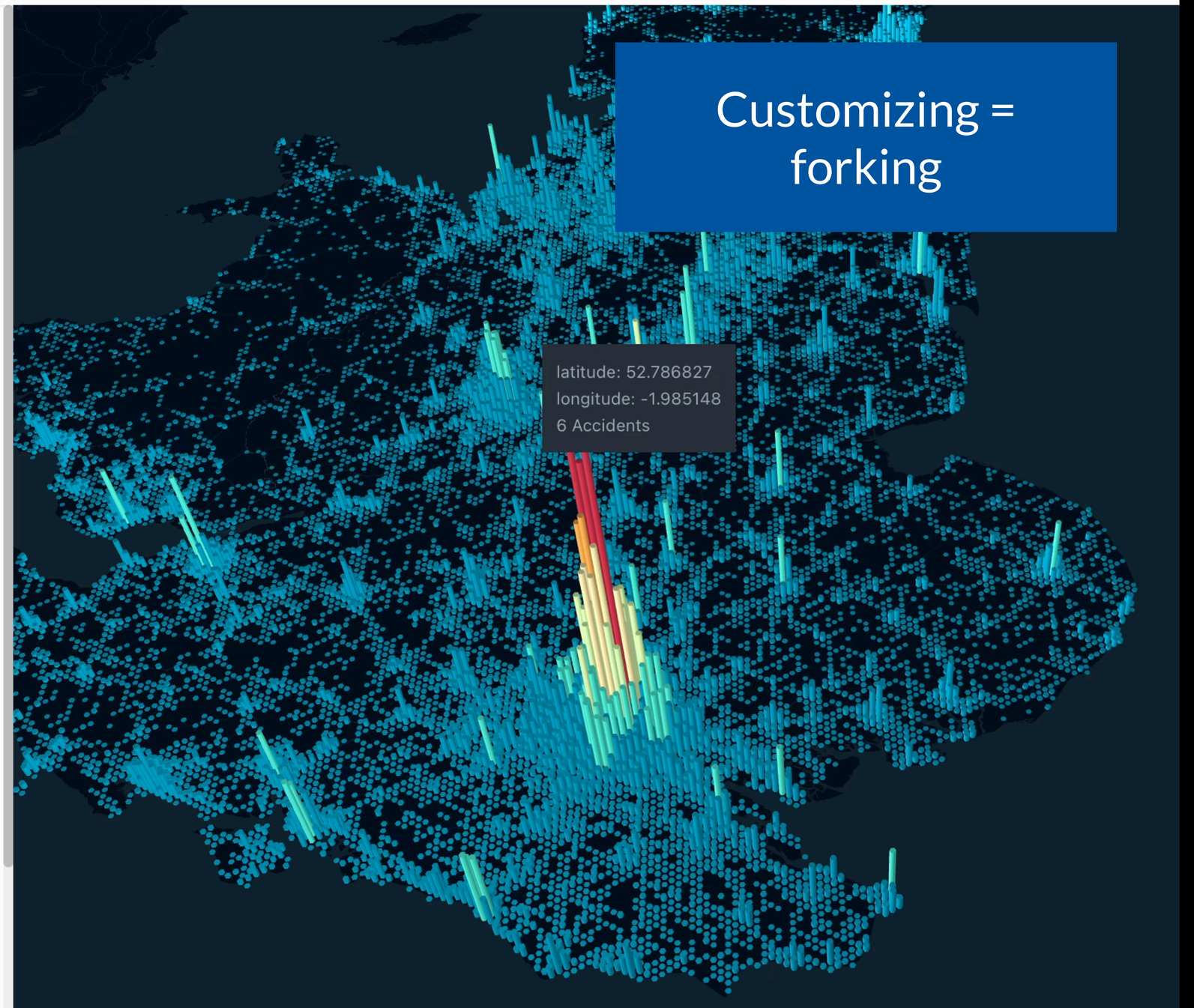
or $2,075/mo

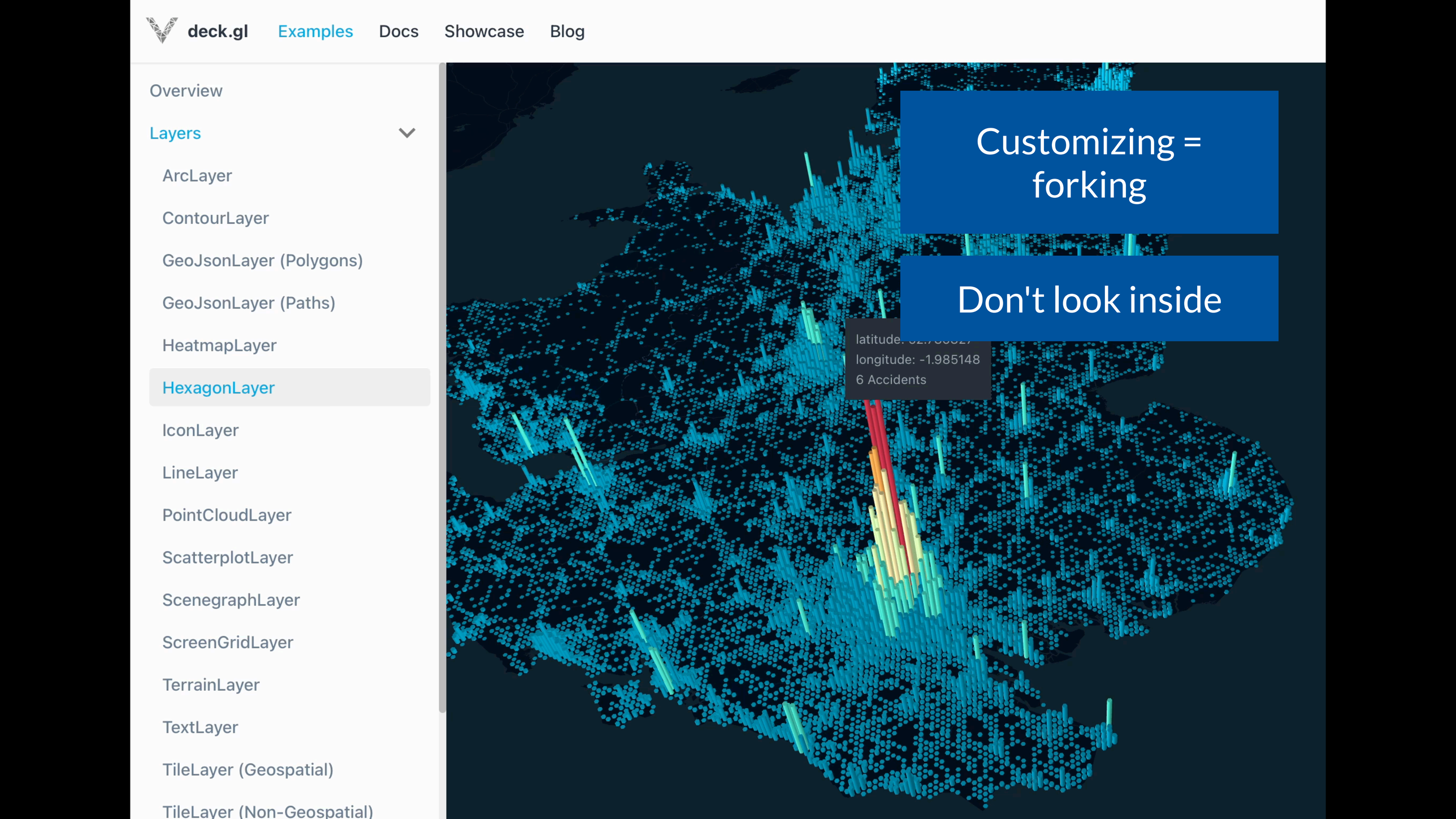
Includes installation estimate

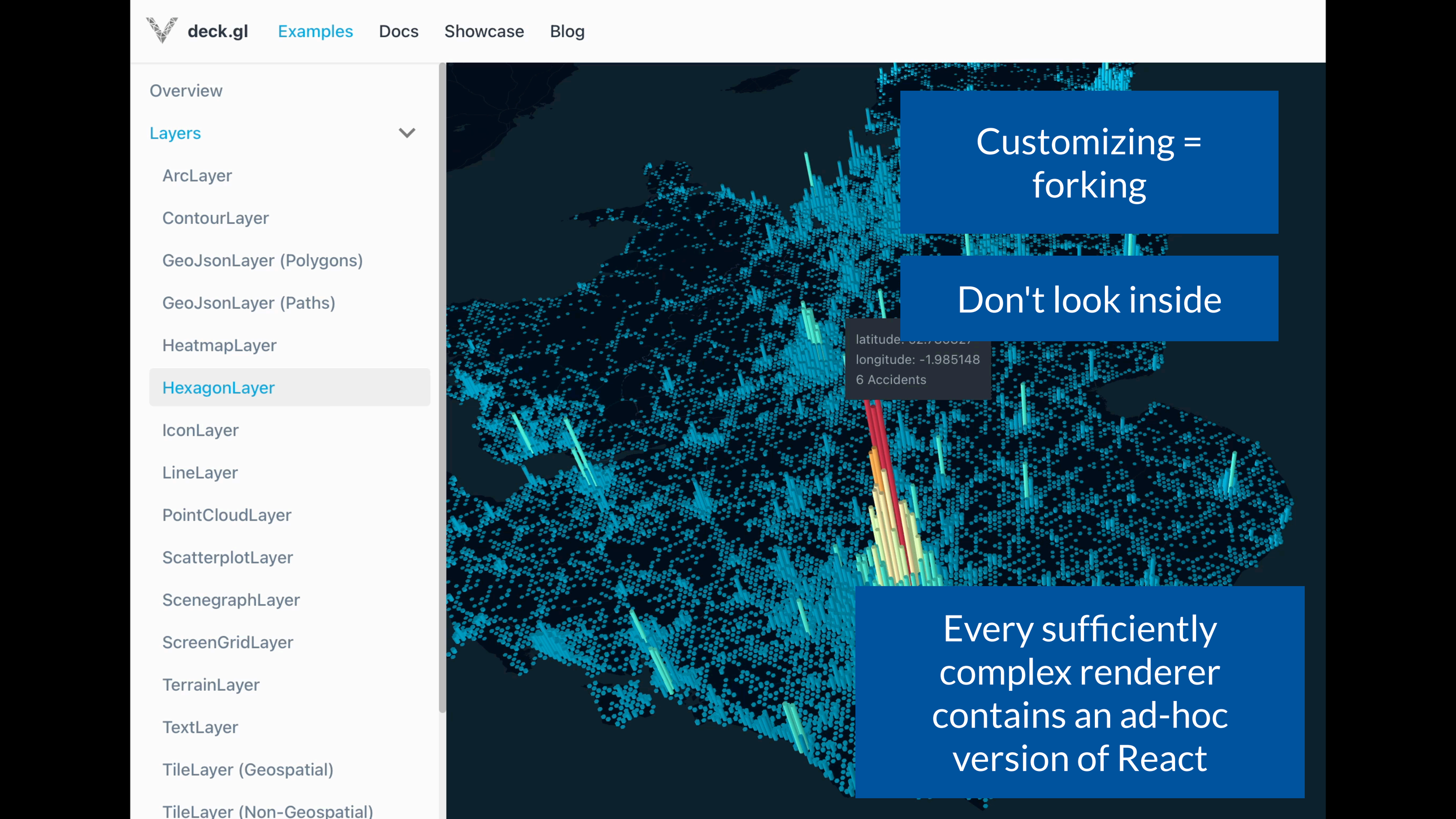
Confirm availability →

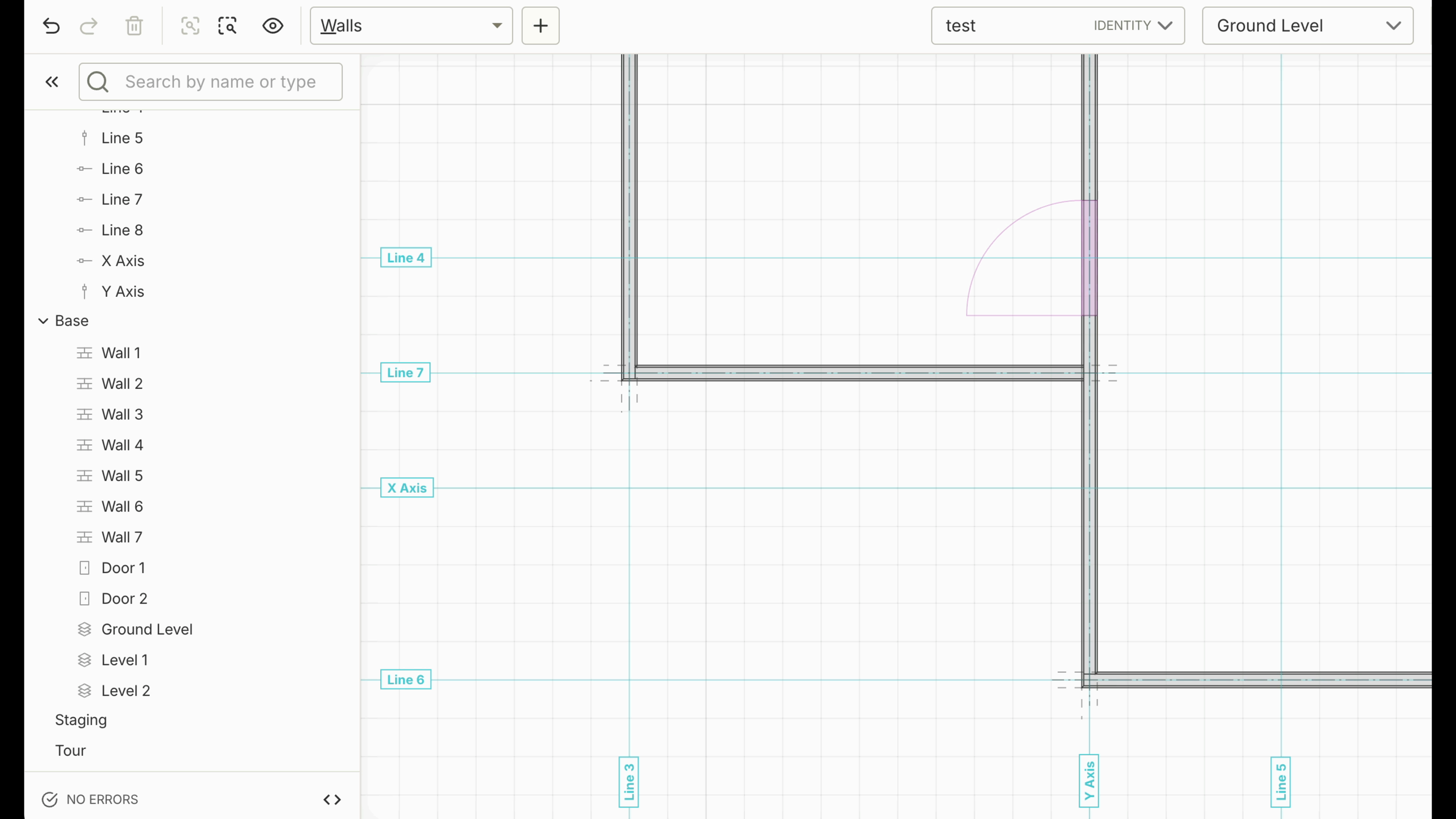


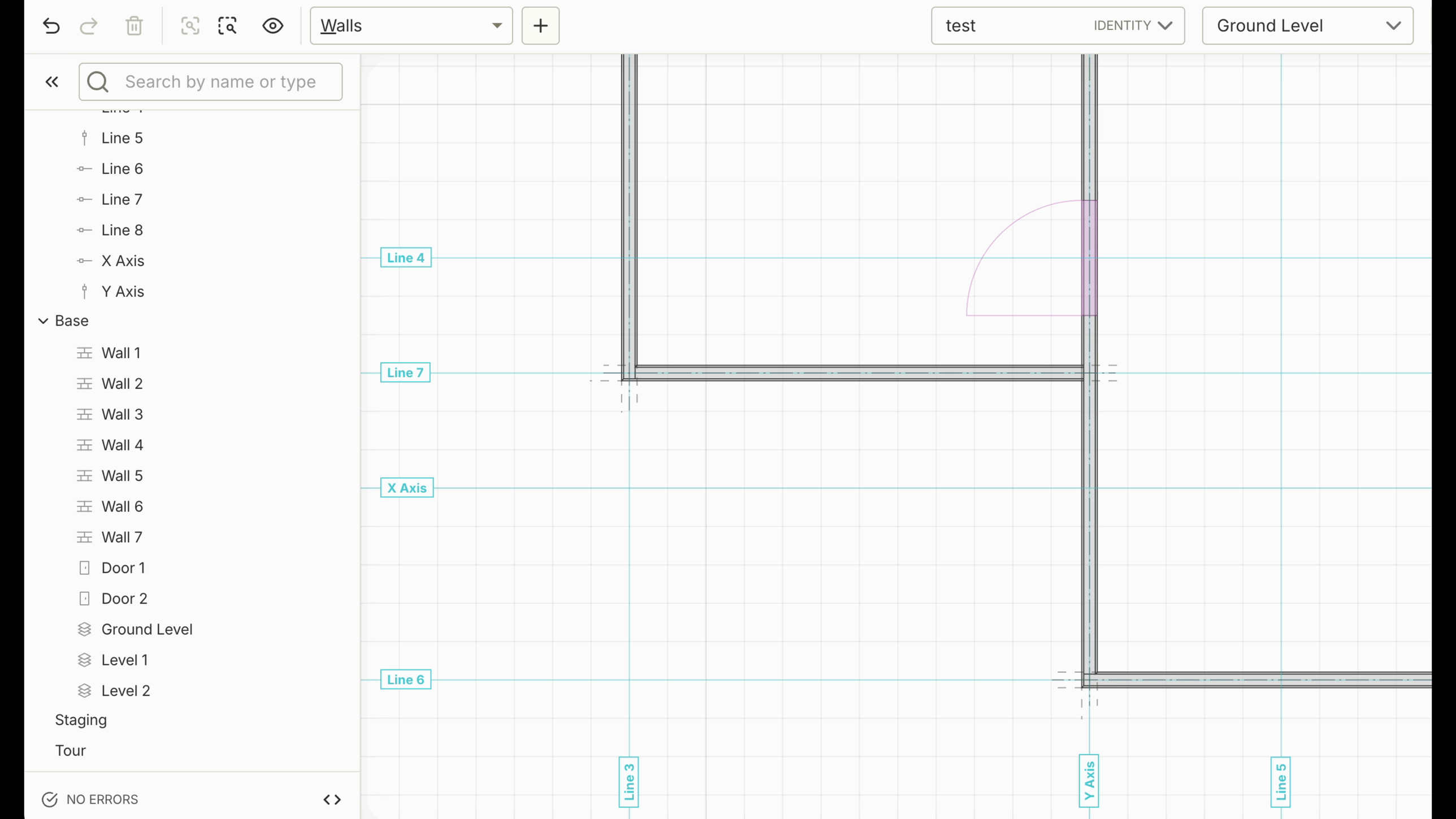


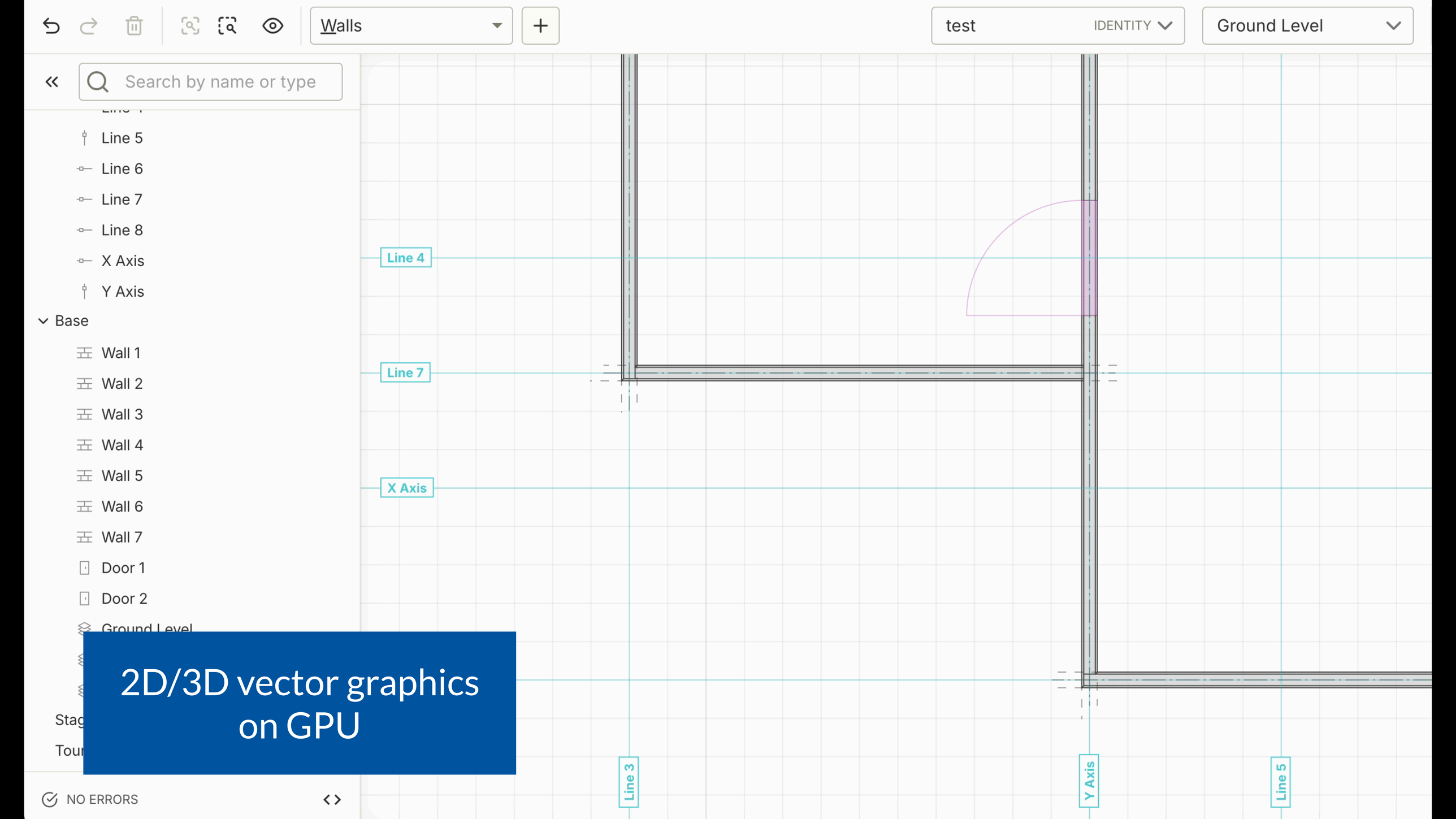


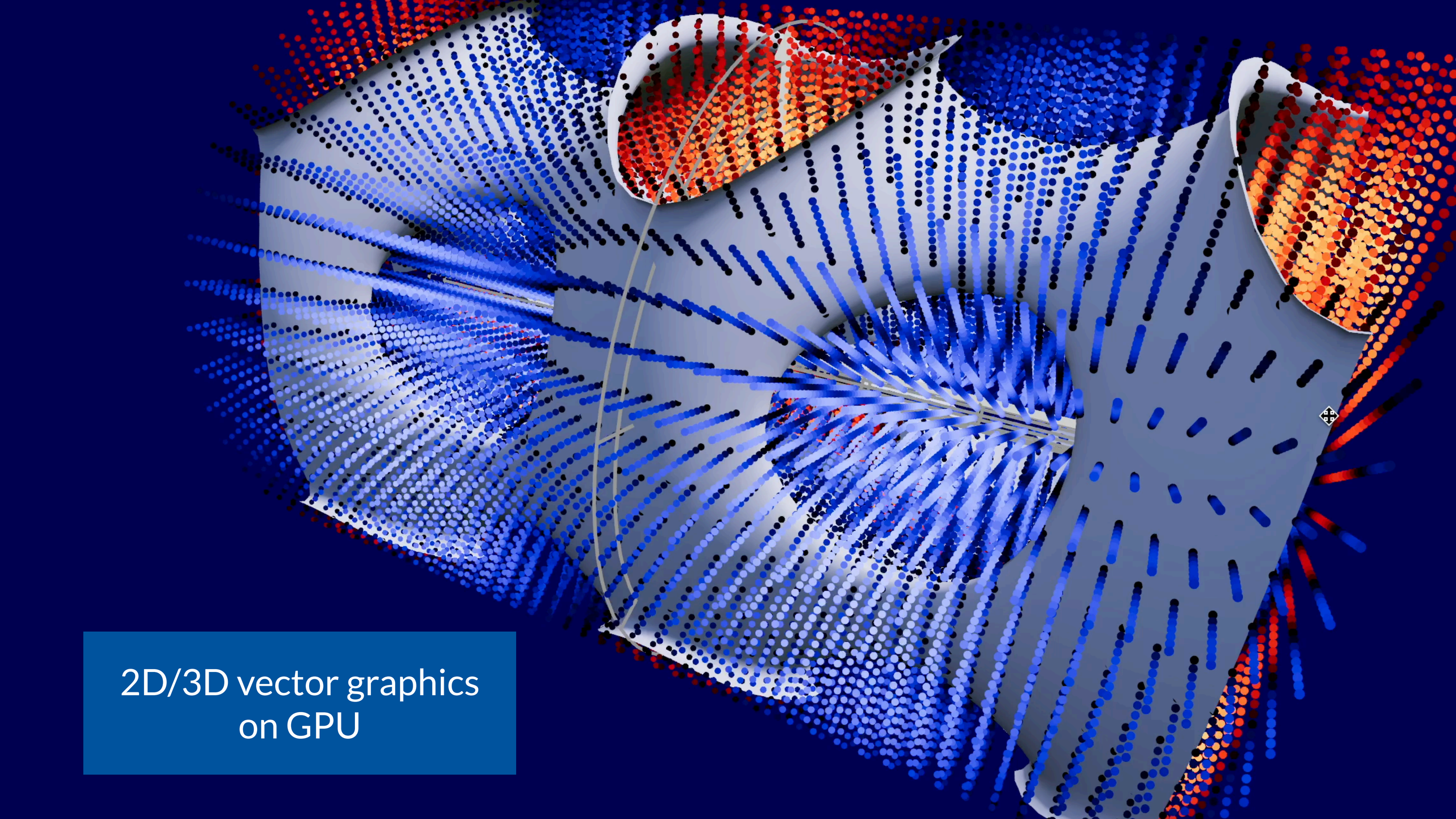

deck.gl
Examples
Docs
Showcase
Blog
Overview
Layers
ArcLayer
ContourLayer
GeoJsonLayer (Polygons)
GeoJsonLayer (Paths)
HeatmapLayer
HexagonLayer
IconLayer
LineLayer
PointCloudLayer
ScatterplotLayer
ScenegraphLayer
ScreenGridLayer
TerrainLayer
TextLayer
TileLayer (Geospatial)
TileLayer (Non-Geospatial)
Customizing = forking
Don't look inside
longitude: -1.985148
6 Accidents


deck.gl
Examples
Docs
Showcase
Blog
Overview
Layers
ArcLayer
ContourLayer
GeoJsonLayer (Polygons)
GeoJsonLayer (Paths)
HeatmapLayer
HexagonLayer
IconLayer
LineLayer
PointCloudLayer
ScatterplotLayer
ScenegraphLayer
ScreenGridLayer
TerrainLayer
TextLayer
TileLayer (Geospatial)
TileLayer (Non-Geospatial)
longitude: -1.985148
6 Accidents
Customizing = forking
Don't look inside
Every sufficiently complex renderer contains an ad-hoc version of React

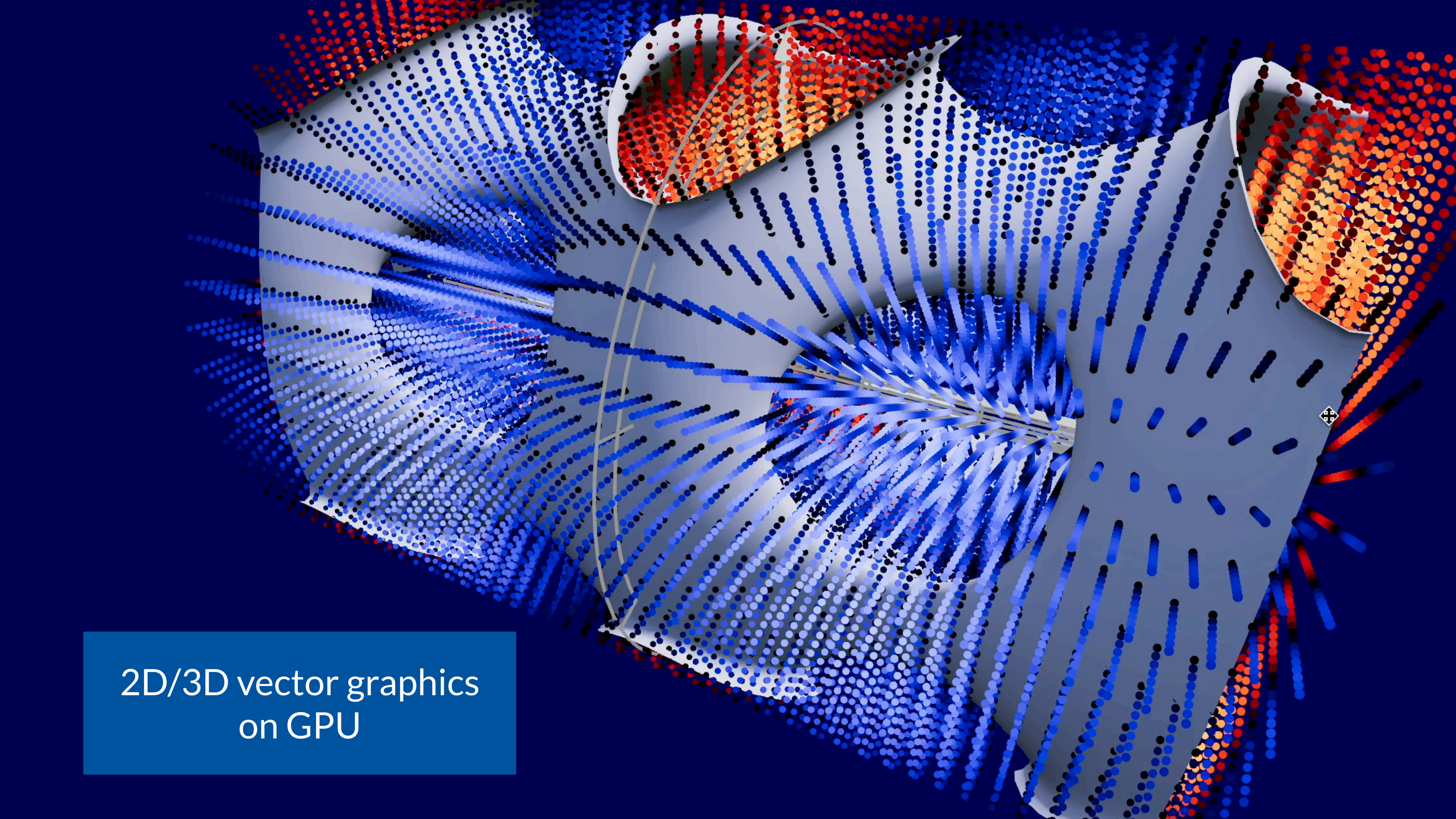

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
Line 4
Line 7
X Axis
Line 6
Line 3
Y Axis
Line 5


Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
Line 4
Line 7
X Axis
Line 6
Line 3
Y Axis
Line 5


Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Line 4
Line 7
X Axis
Line 3
Y Axis
Line 5
NO ERRORS
2D/3D vector graphics on GPU



2D/3D vector graphics
on GPU



2D/3D vector graphics
on GPU

"We want a ...

"We want a ...

- 3D house/car configurator
- Boutique data visualization
- GPU-driven CAD app

"We want a ...

- 3D house/car configurator
- Boutique data visualization
- GPU-driven CAD app

- With transparency and global illumination

"We want a ...

- 3D house/car configurator
- Boutique data visualization
- GPU-driven CAD app

- With transparency and global illumination

... can you build it?"

"We want a ...

"Yes, I can ...

- 3D house/car configurator
- Boutique data visualization
- GPU-driven CAD app

- With transparency and global illumination

... can you build it?"

"We want a ...

- 3D house/car configurator
- Boutique data visualization
- GPU-driven CAD app

- With transparency and global illumination

"Yes, I can ...

But **GPU** will take
N weeks/months/years

... can you build it?"

"We want a ...

- 3D house/car configurator
- Boutique data visualization
- GPU-driven CAD app

- With transparency and global illumination

"Yes, I can ...

But **GPU** will take
N weeks/months/years

...and you **won't**
be able to maintain it"

... can you build it?"

"We want a ...

- 3D house/car configurator
- Boutique data visualization
- GPU-driven CAD app

- With transparency and global illumination

"Yes, I can ...

But **GPU** will take
N weeks/months/years

...and you **won't**
be able to maintain it"

Why not?

... can you build it?"

This Talk

This Talk

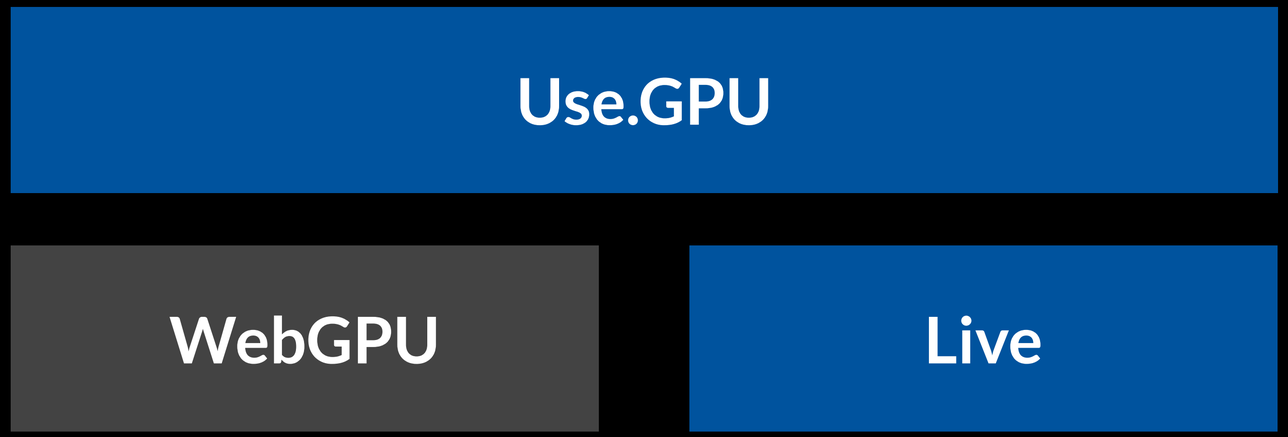
Use.GPU

This Talk

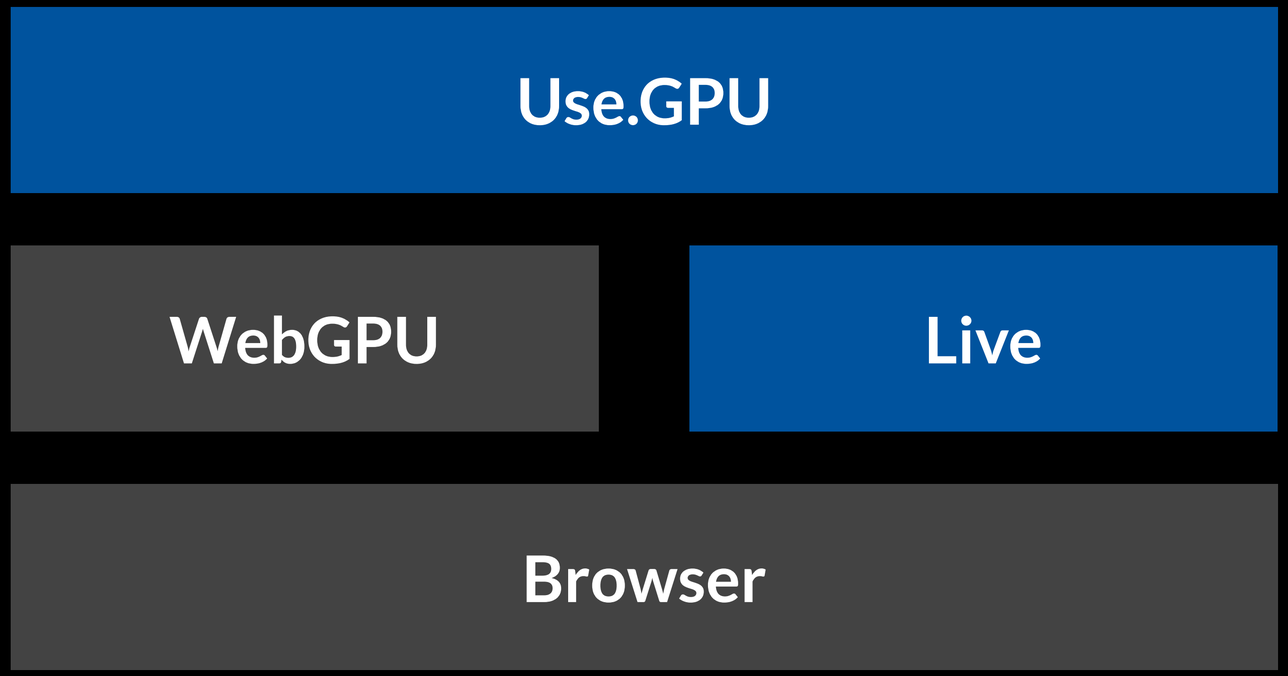
Use.GPU

WebGPU

This Talk

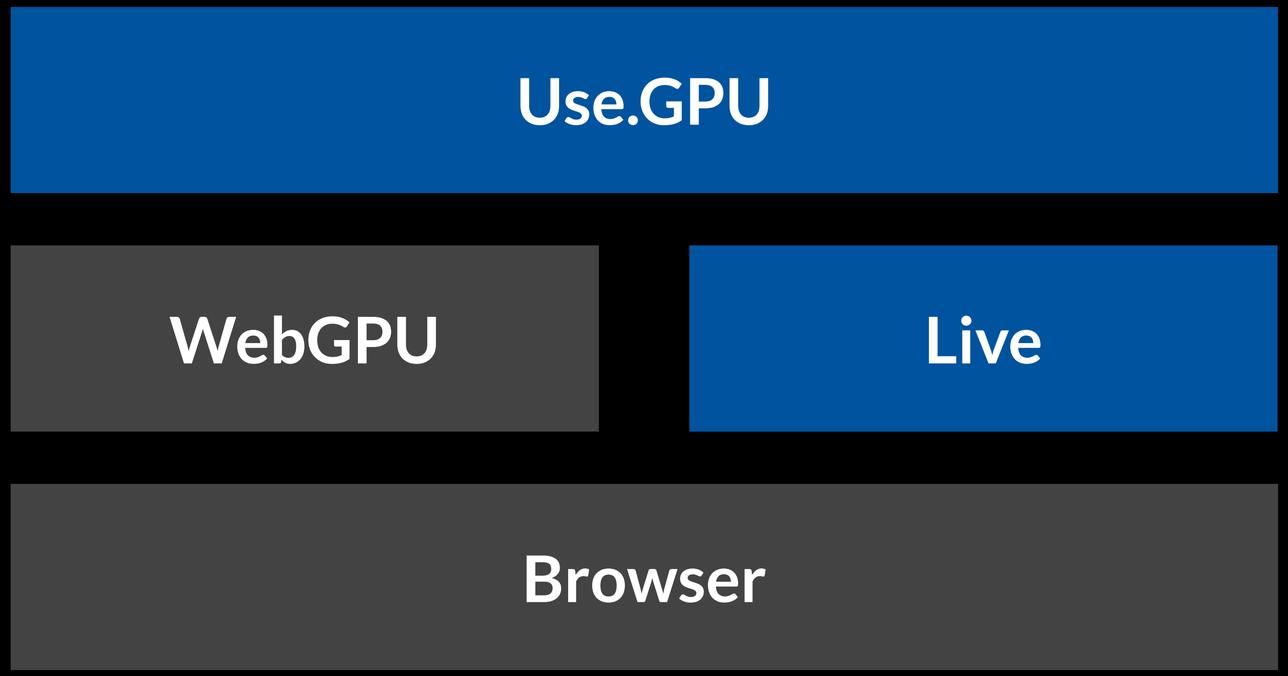


This Talk



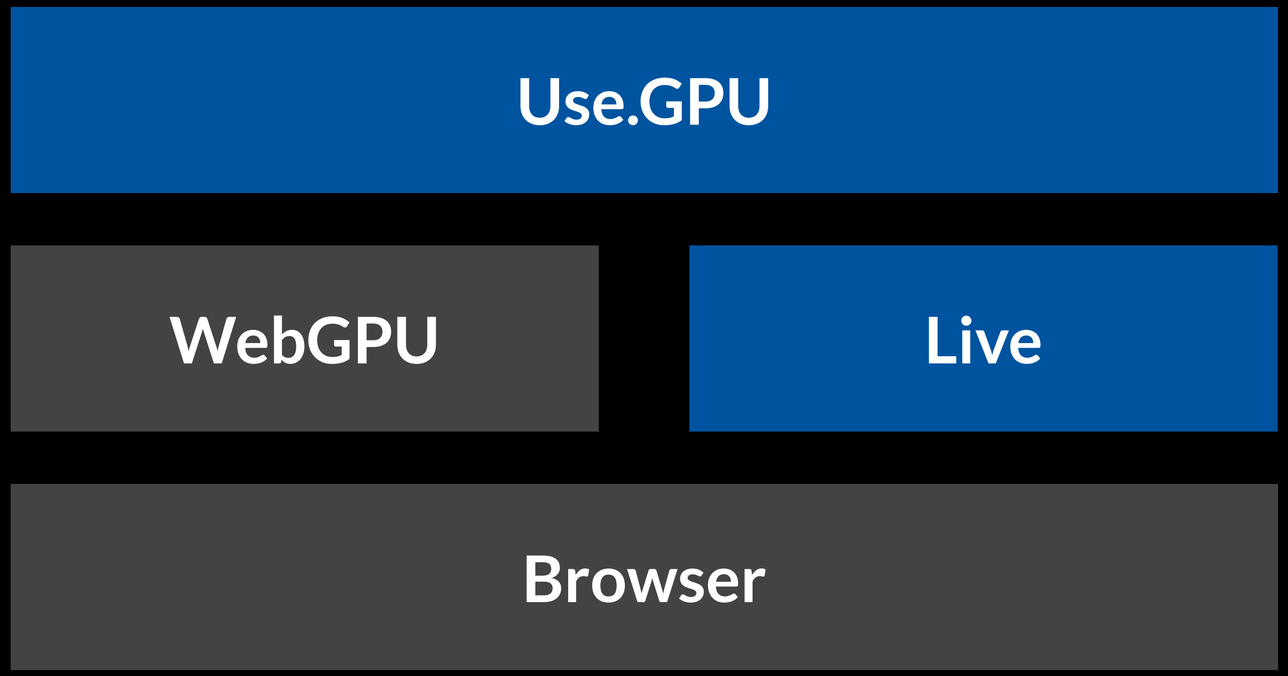
This Talk

- Incrementalism



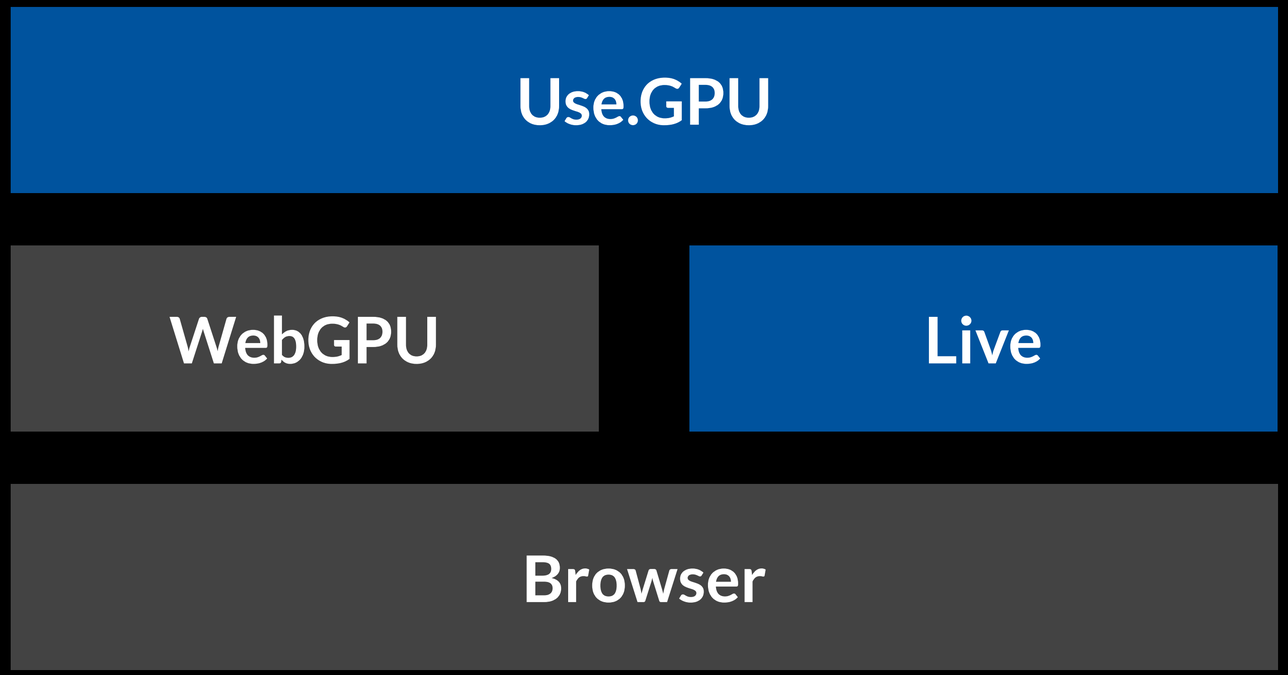
This Talk

- Incrementalism
- Graphics vs Web



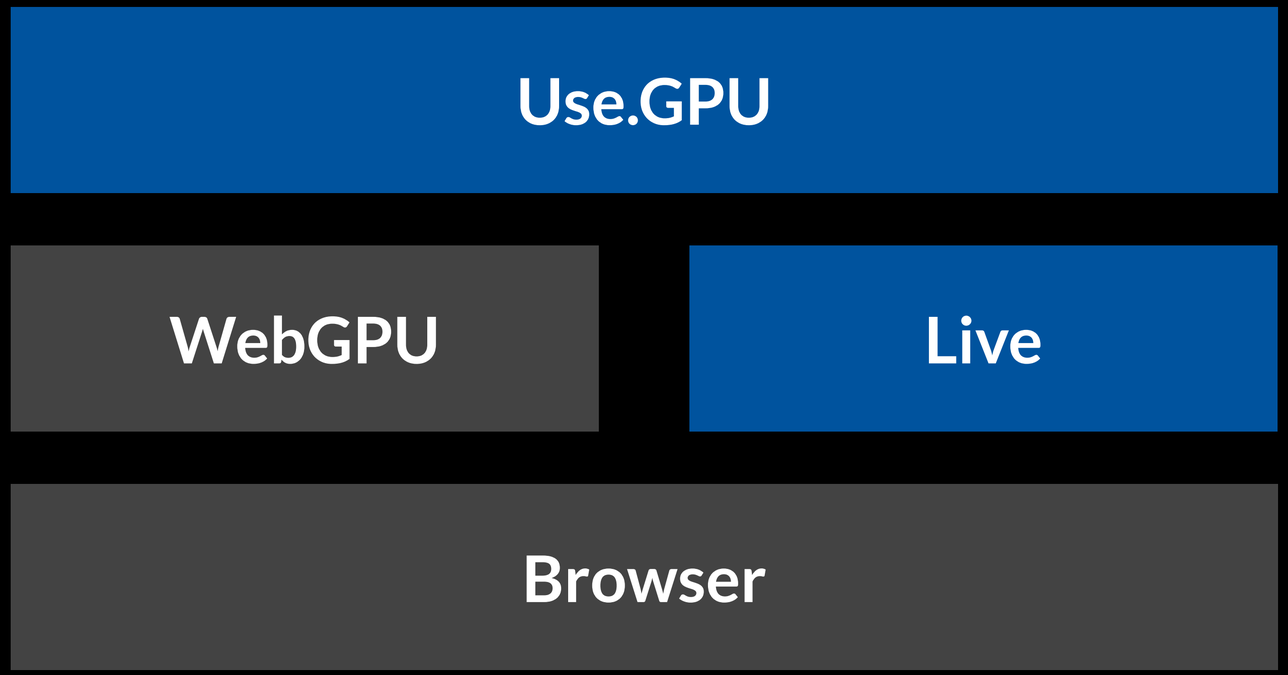
This Talk

- Incrementalism
- Graphics vs Web
- What went wrong with GPUs



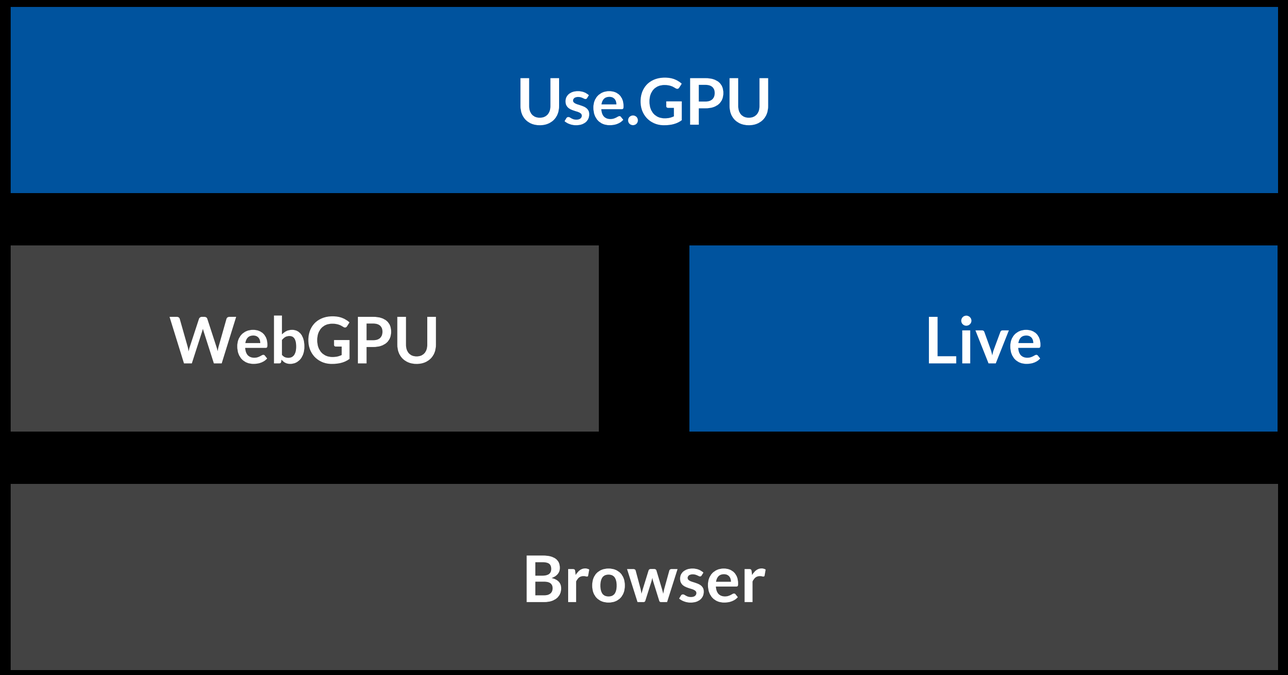
This Talk

- Incrementalism
- Graphics vs Web
- What went wrong with GPUs
- Live run-time



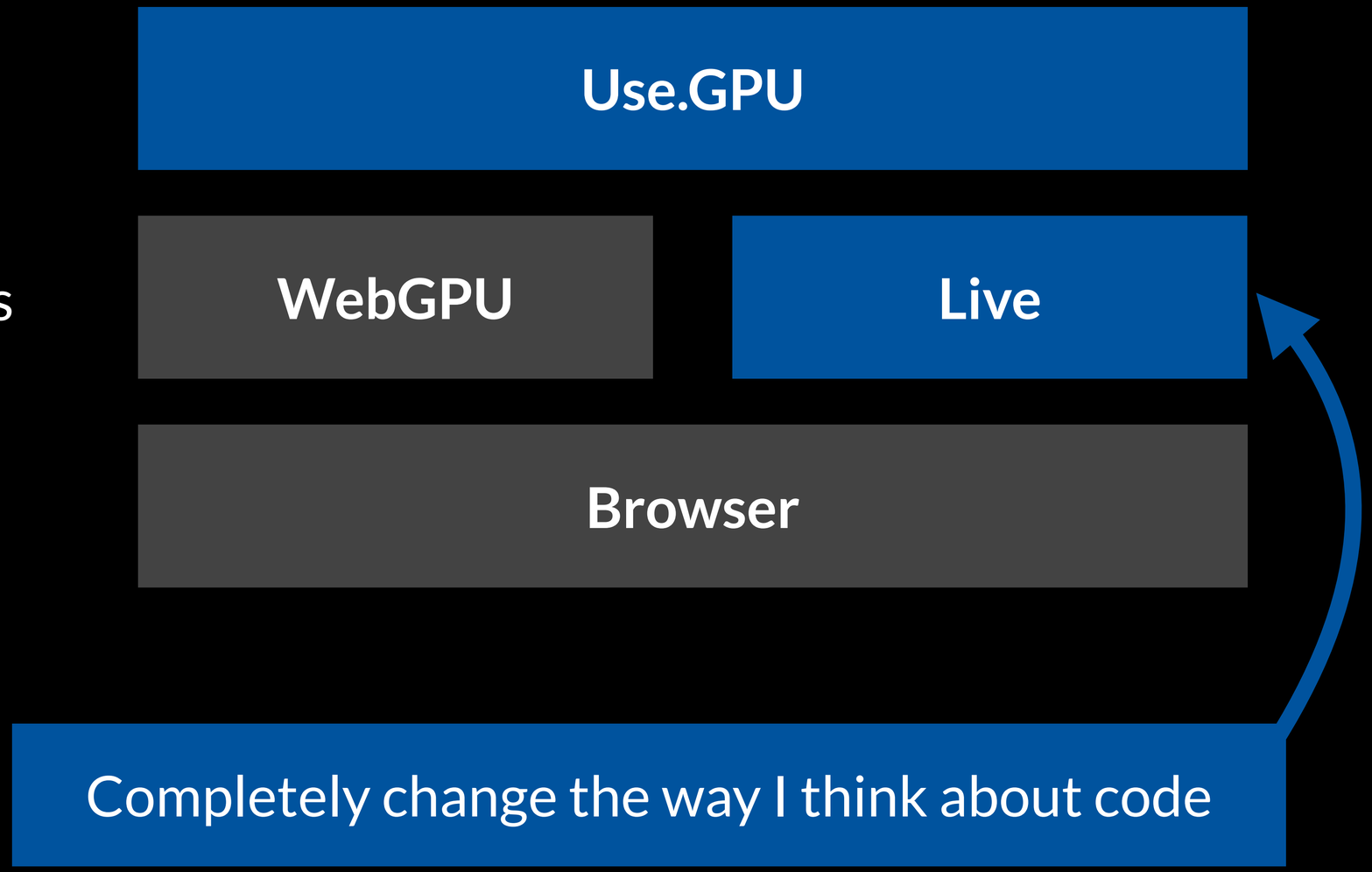
This Talk

- Incrementalism
- Graphics vs Web
- What went wrong with GPUs
- Live run-time
- Use.GPU components

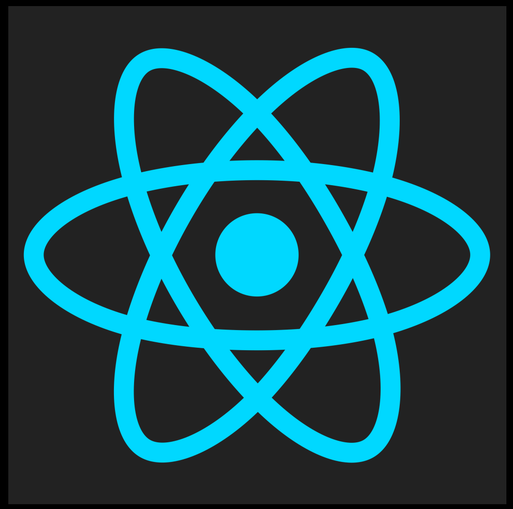


This Talk

- Incrementalism
- Graphics vs Web
- What went wrong with GPUs
- Live run-time
- Use.GPU components

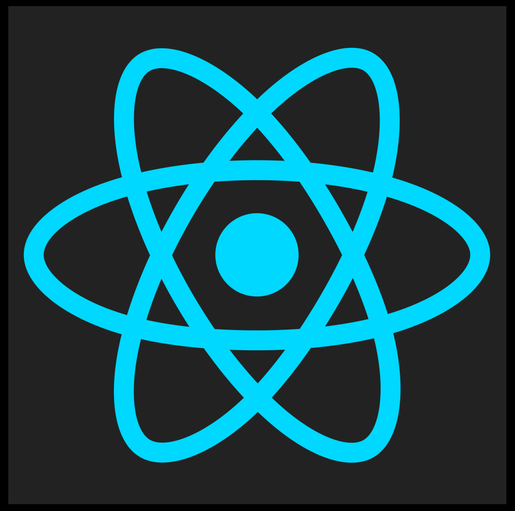


Live = alt-React



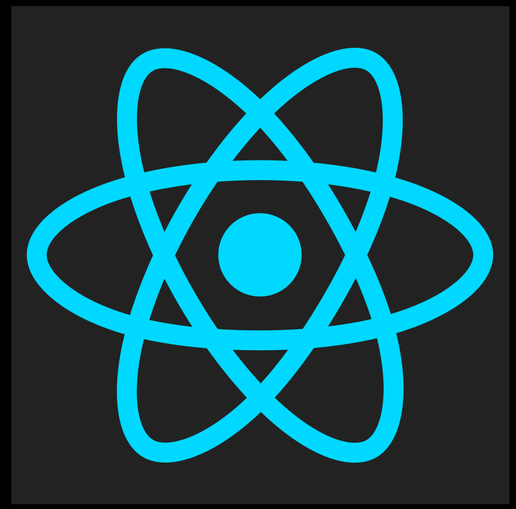
Live = alt-React

React clone



Live = alt-React

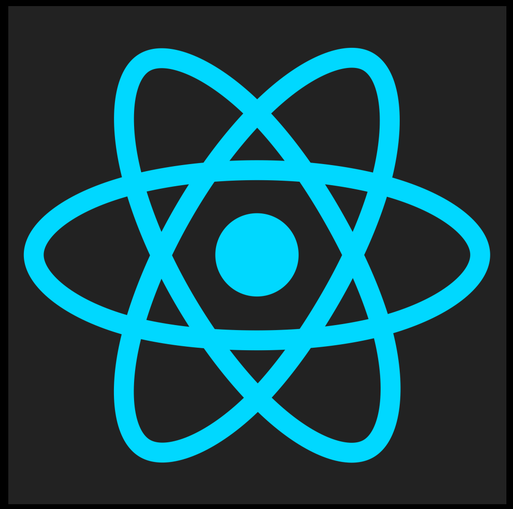
React clone



- **Incremental**
  Avoid redundant recomputation

Live = alt-React

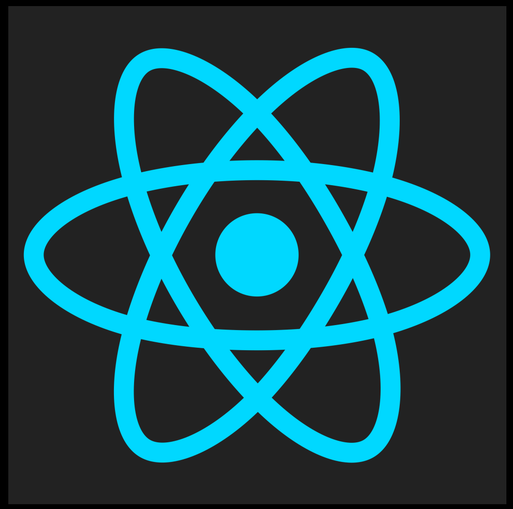
React clone



- **Incremental**
  Avoid redundant recomputation
- **Reactive**
  Dispatch and data flow is implicit and 1-way

Live = alt-React

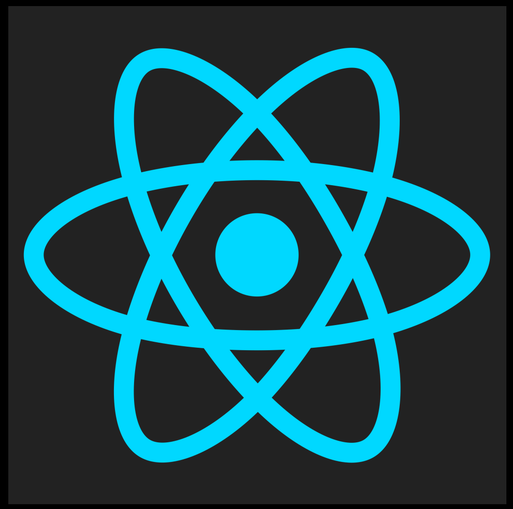
React clone



- **Incremental**
  Avoid redundant recomputation
- **Reactive**
  Dispatch and data flow is implicit and 1-way
- **Declarative**
  Side-effects are auto-mounted and disposed

Live = alt-React

React clone

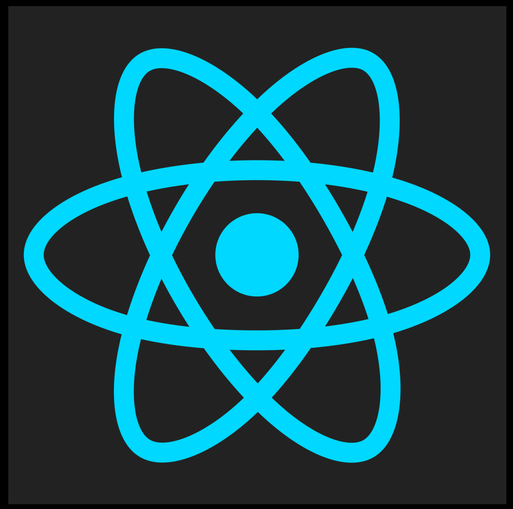


- **Incremental**
  Avoid redundant recomputation
- **Reactive**
  Dispatch and data flow is implicit and 1-way
- **Declarative**
  Side-effects are auto-mounted and disposed

React did something **very interesting**

Live = alt-React

React clone



I teach devs React

- **Incremental**
  Avoid redundant recomputation
- **Reactive**
  Dispatch and data flow is implicit and 1-way
- **Declarative**
  Side-effects are auto-mounted and disposed

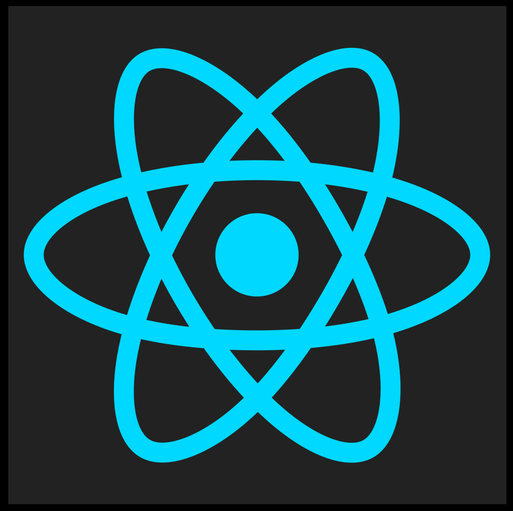
React did something **very interesting**

Live = alt-React

React clone

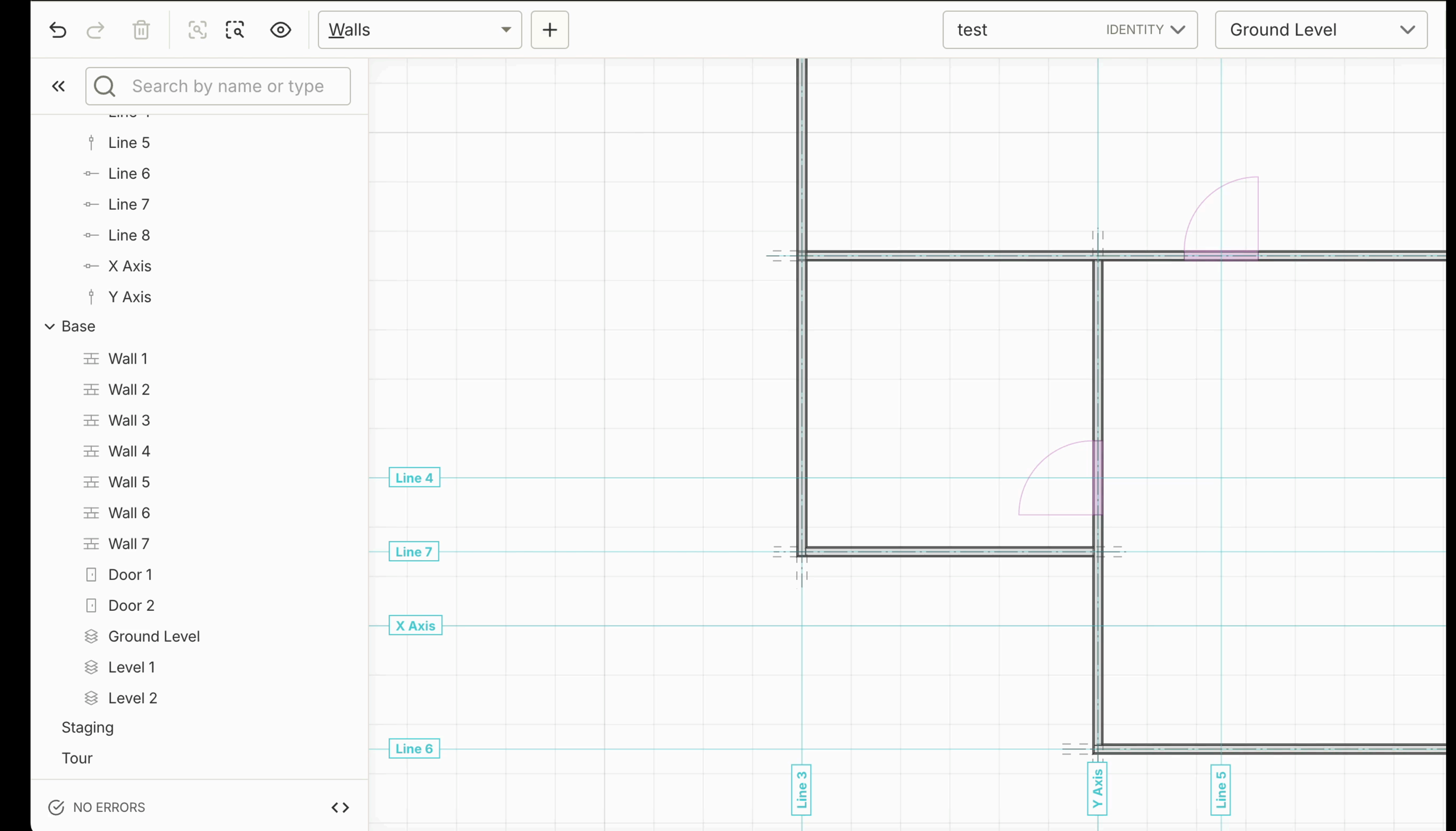
- **Incremental**
  Avoid redundant recomputation
- **Reactive**
  Dispatch and data flow is implicit and 1-way
- **Declarative**
  Side-effects are auto-mounted and disposed

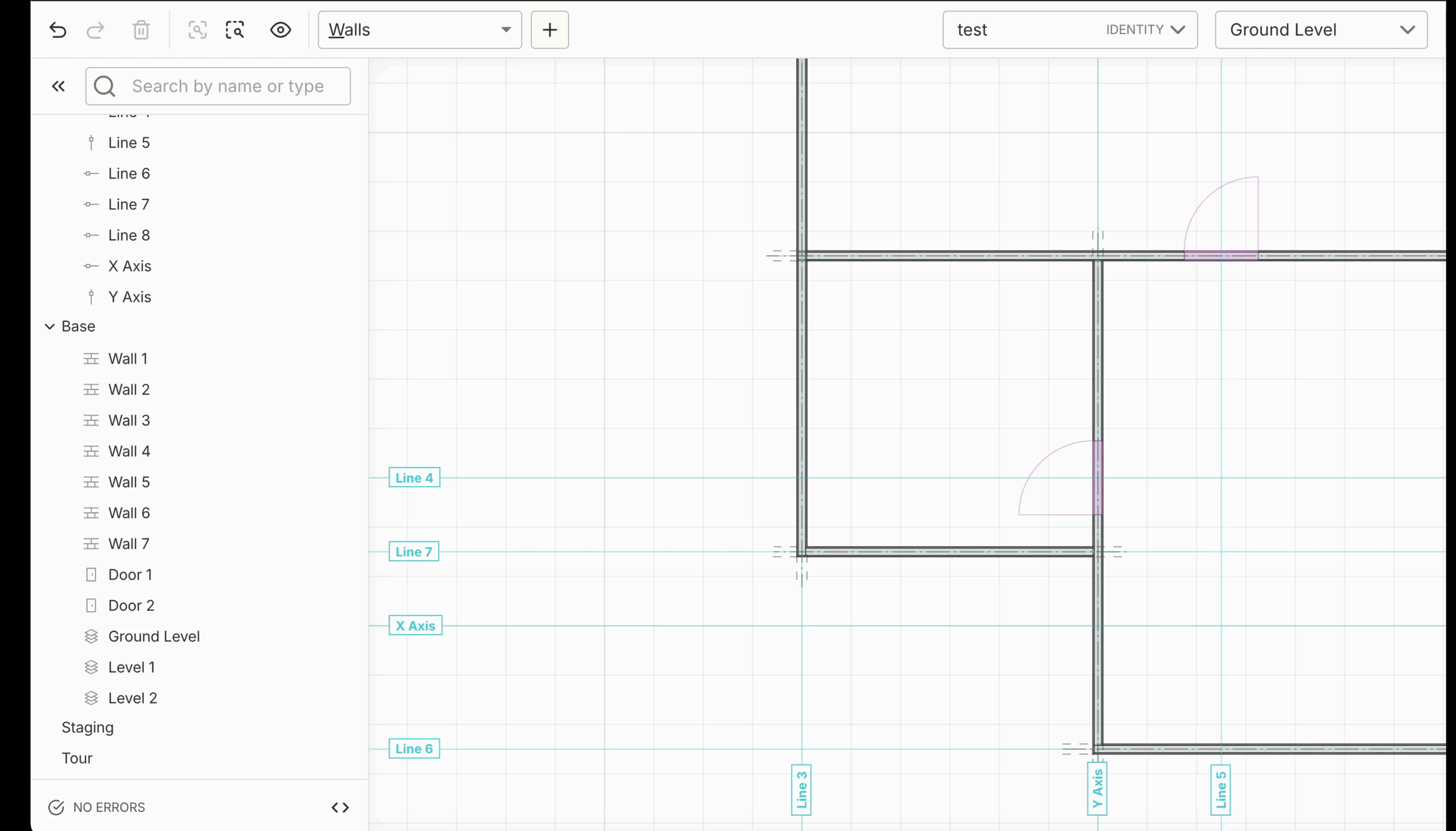
I teach devs React

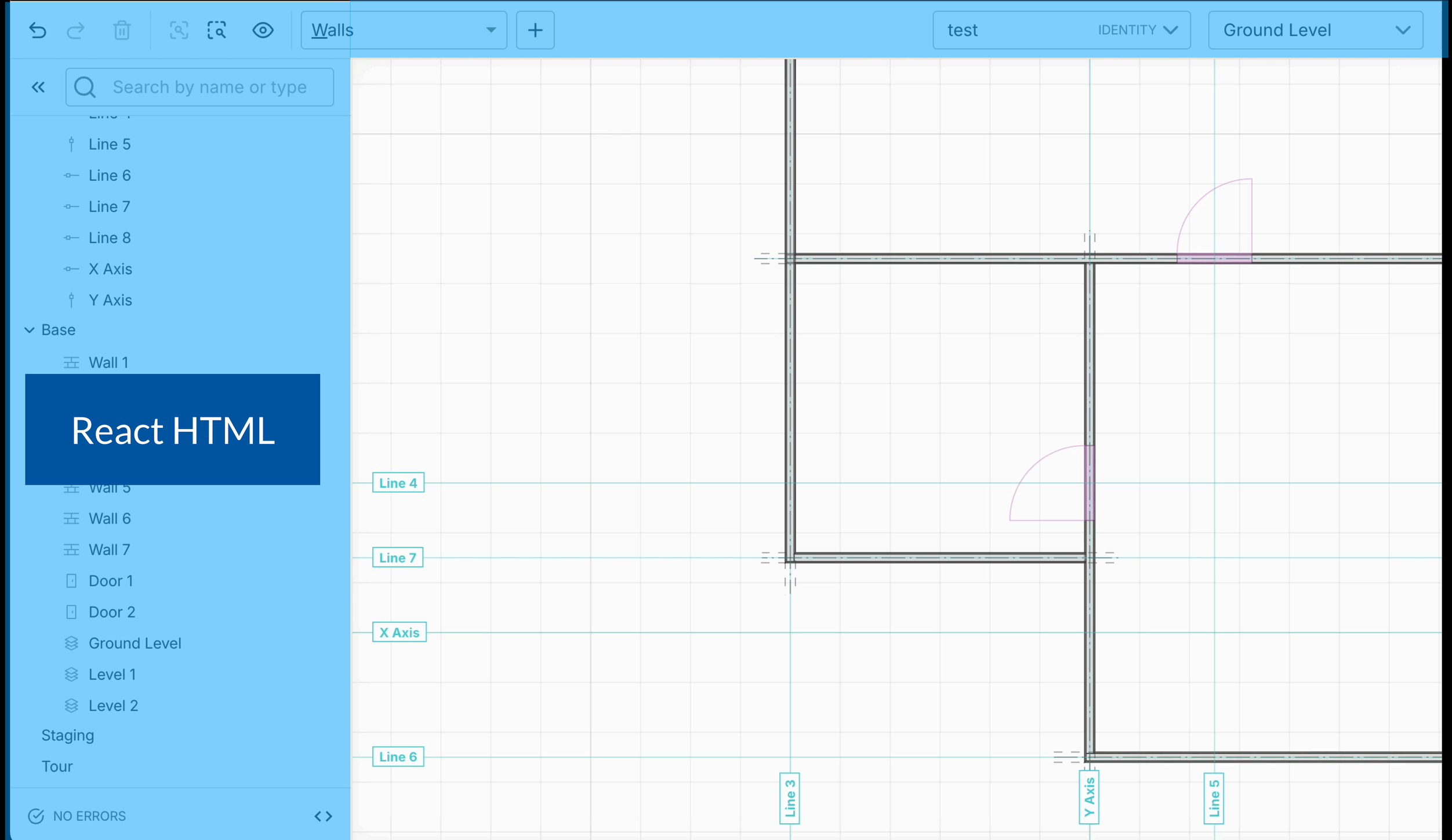


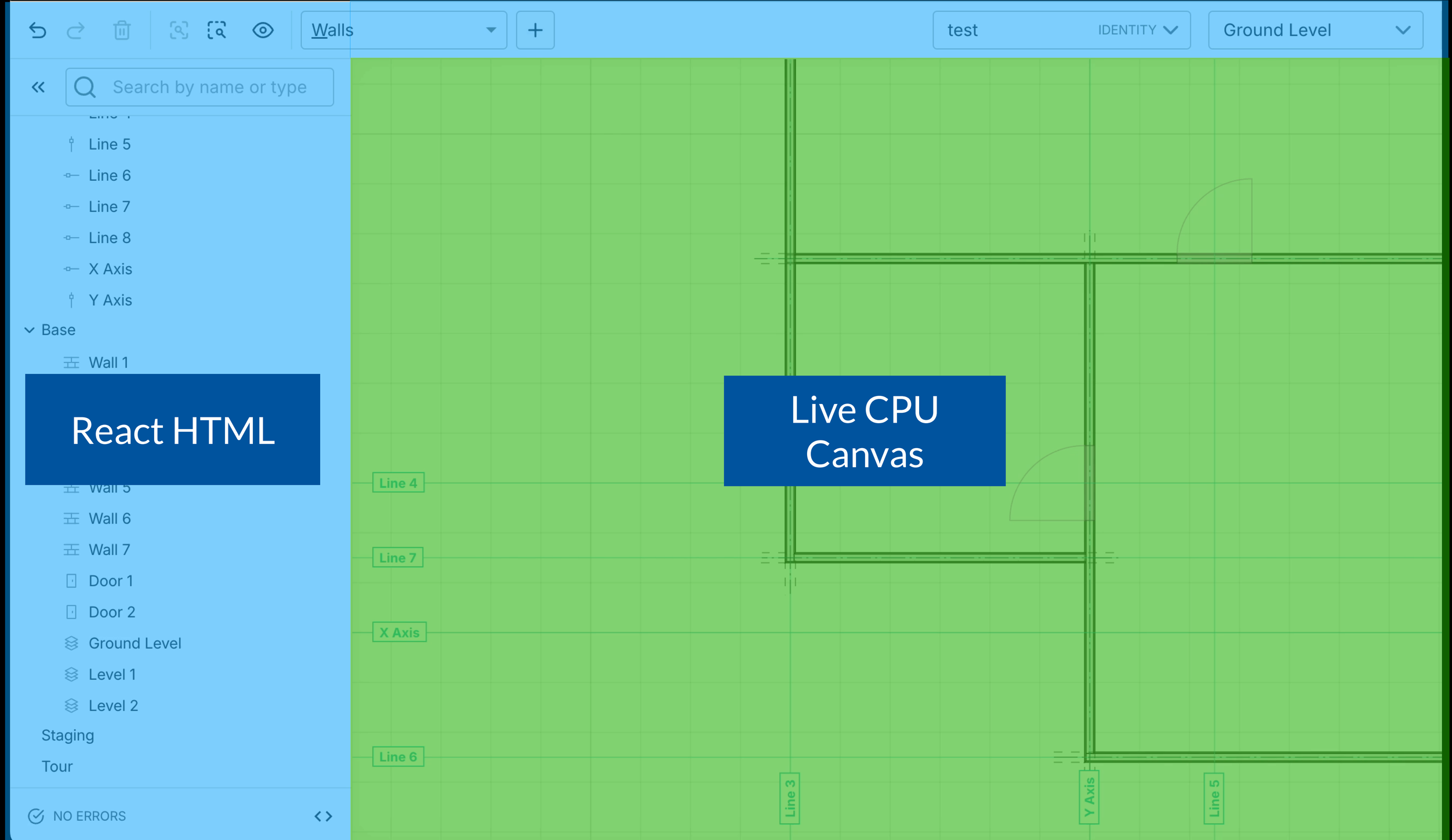
Until you build
Excel or Figma in React,
you don't understand React.

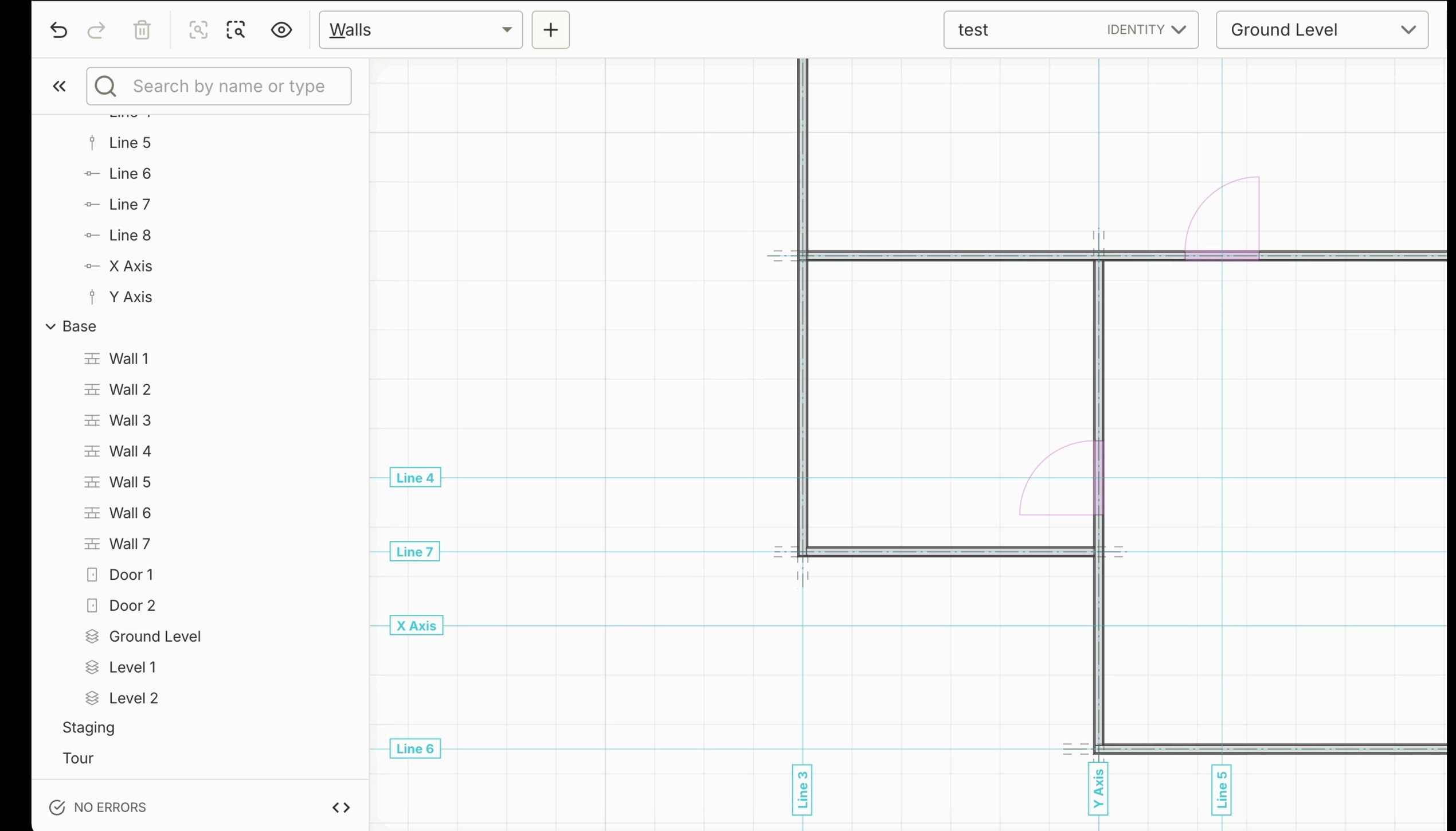
React did something **very interesting**

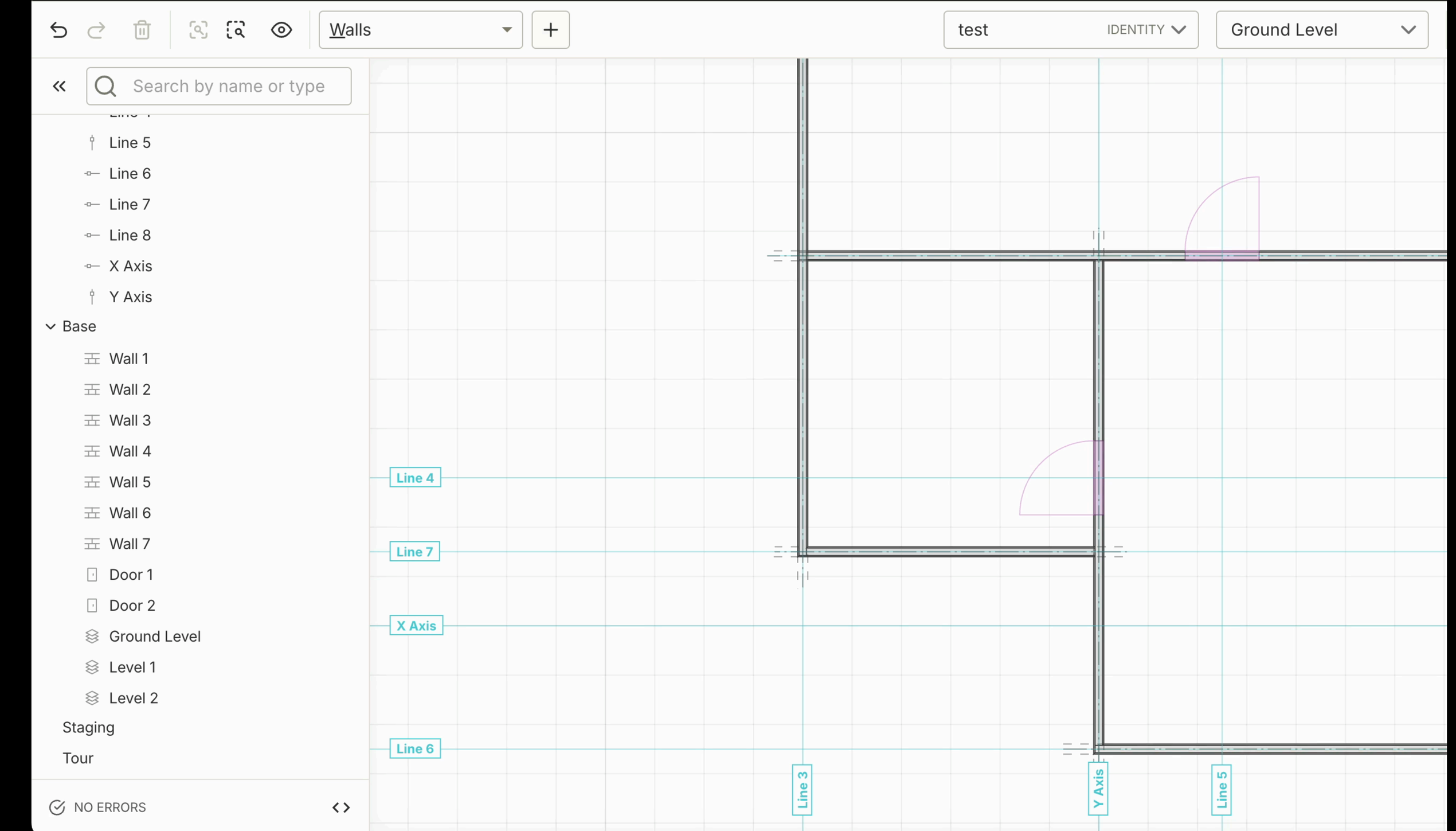

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
Line 4
Line 7
X Axis
Line 6
Line 3
Y Axis
Line 5

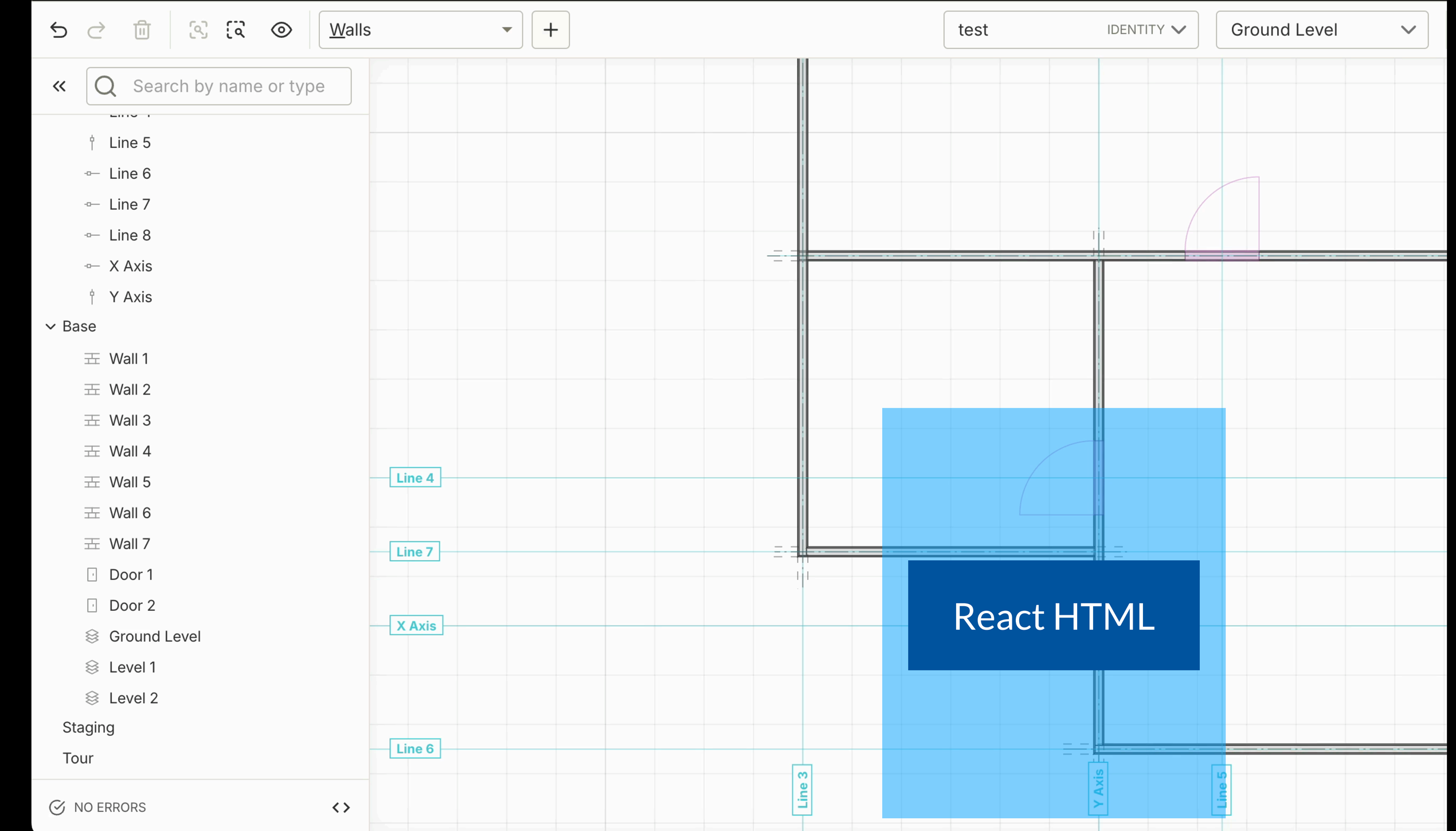

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
Line 4
Line 7
X Axis
Line 6
Line 3
Y Axis
Line 5

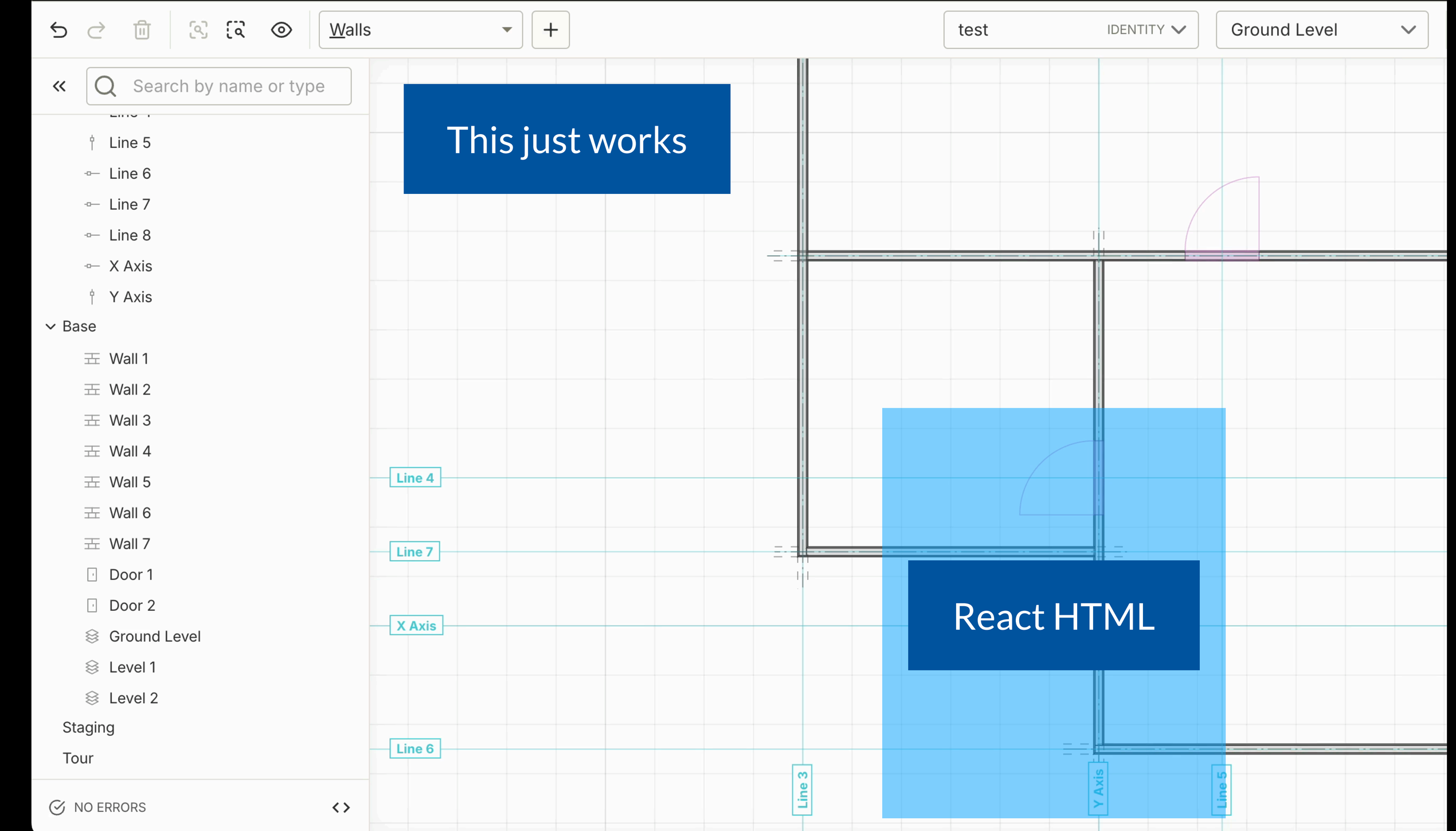

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
React HTML
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
Line 4
Line 7
X Axis
Line 6
Line 3
Y Axis
Line 5


Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
React HTML
Live CPU Canvas
Line 4
Line 7
X Axis
Line 6
Line 3
Y Axis
Line 5


Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
Line 4
Line 7
X Axis
Line 6
Line 3
Y Axis
Line 5


Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
Line 4
Line 7
X Axis
Line 6
Line 3
Y Axis
Line 5


Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
Line 4
Line 7
X Axis
Line 6
Line 3
Y Axis
Line 5
React HTML


Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
This just works
React HTML
Line 4
Line 7
X Axis
Line 6
Line 3
Y Axis
Line 5

Walls

test IDENTITY

Ground Level

Search by name or type

- Line 5
- Line 6
- Line 7
- Line 8
- X Axis
- Y Axis
- Base
  - Wall 1
  - Wall 2
  - Wall 3
  - Wall 4
  - Wall 5
  - Wall 6
  - Wall 7
  - Door 1
  - Door 2
  - Ground Level
  - Level 1
  - Level 2
- Staging
- Tour

NO ERRORS

This just works

React ↔ Live

React HTML

Line 4

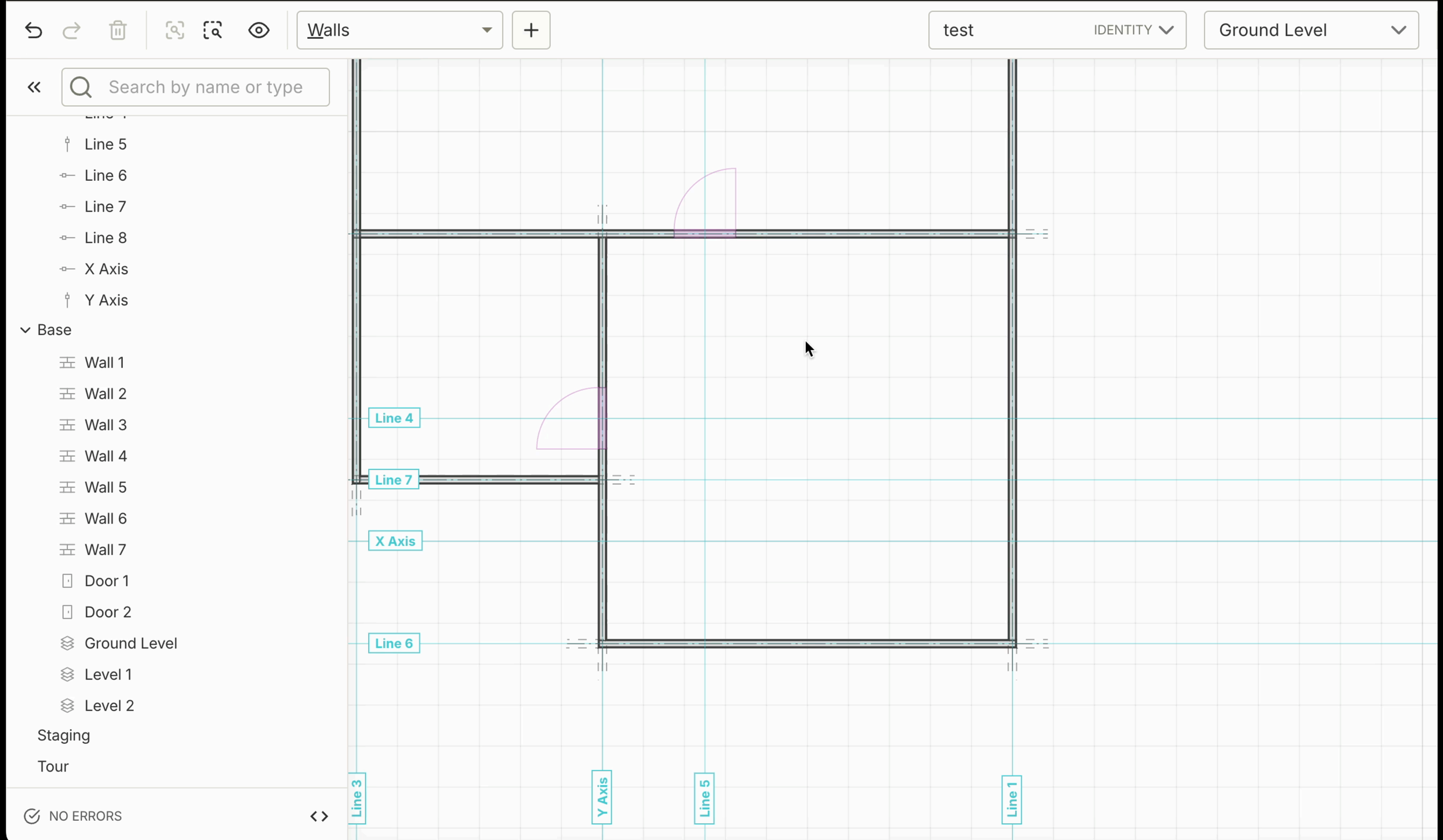
Line 7

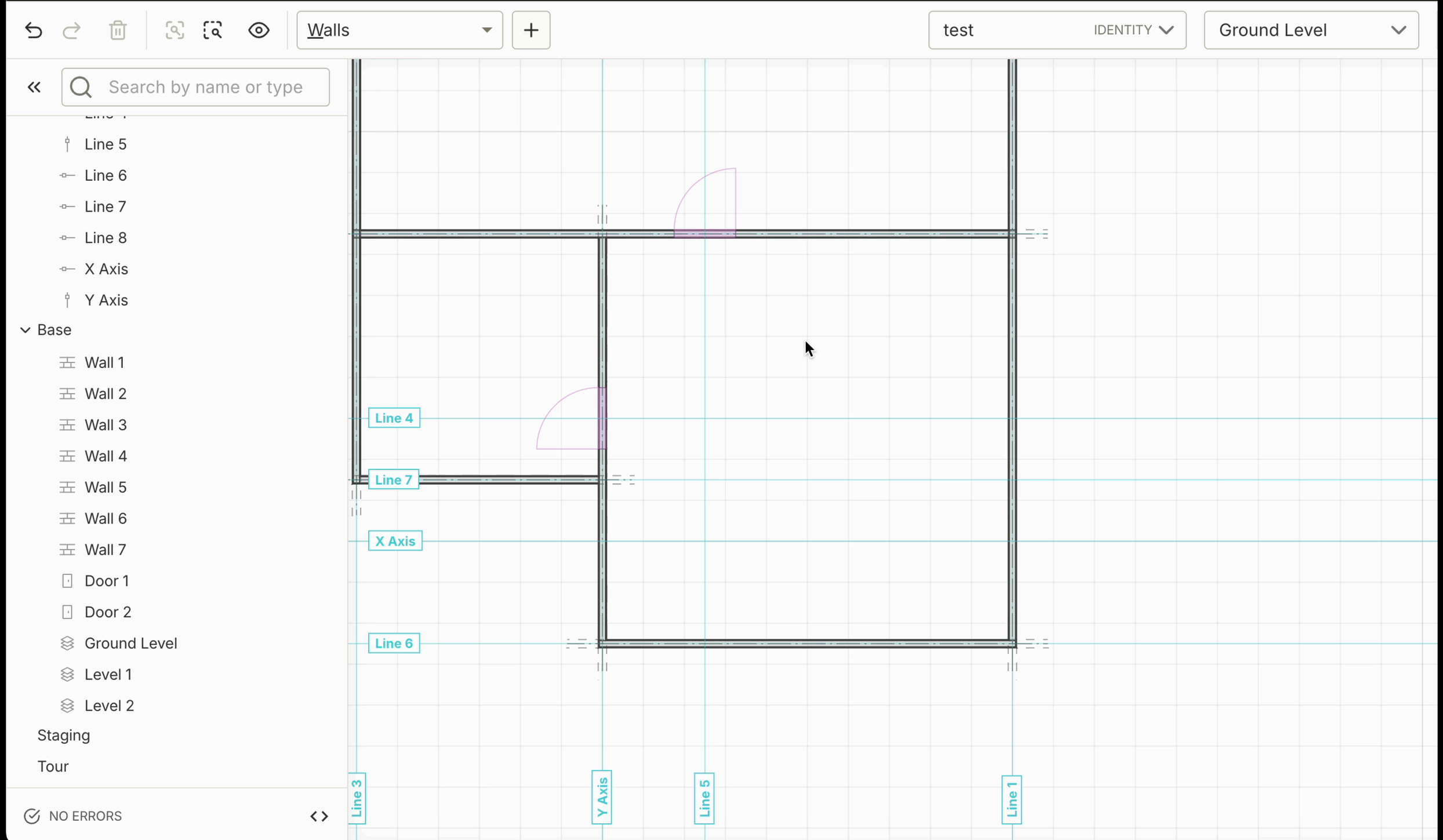
Line 6

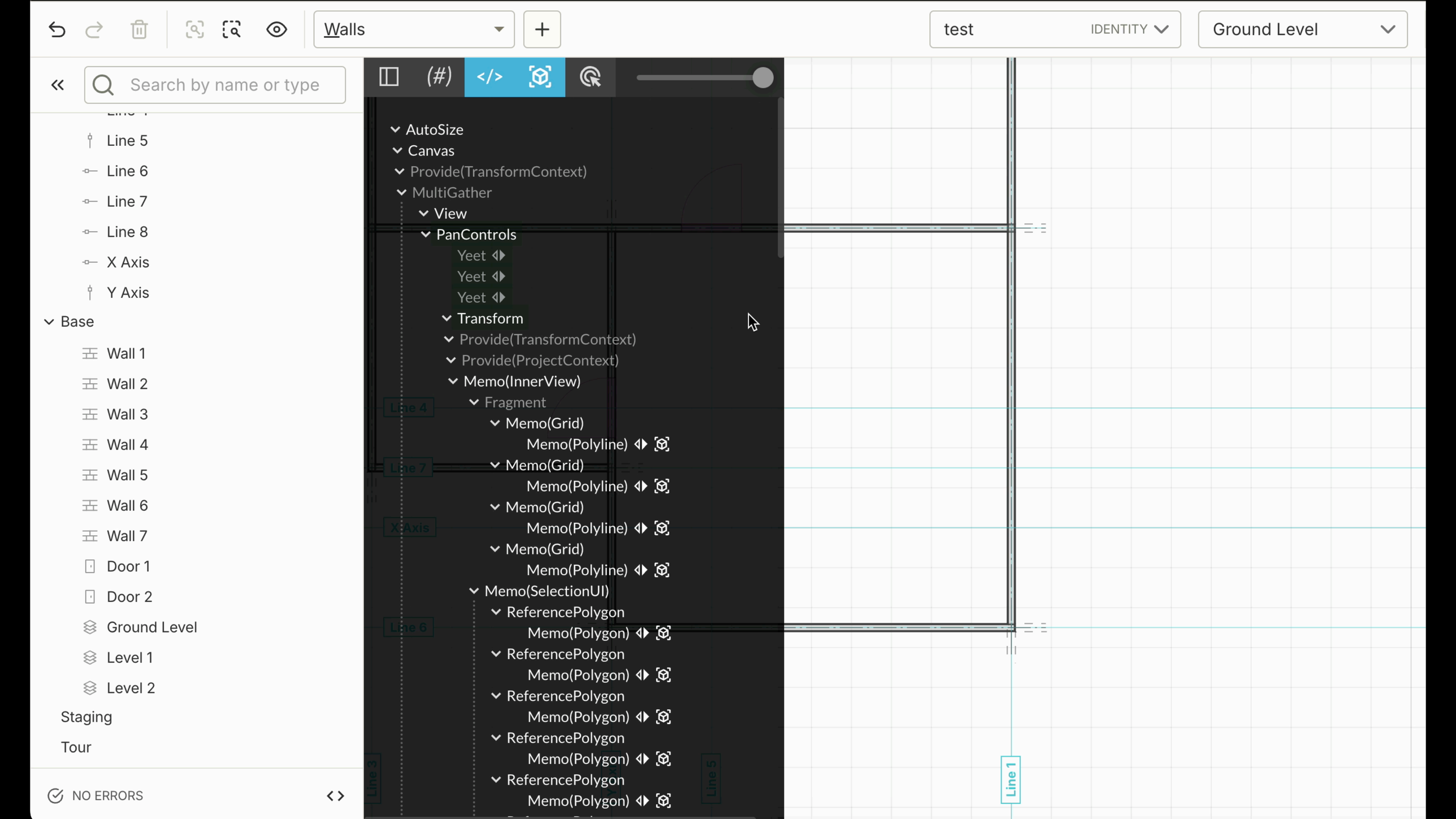
Line 3

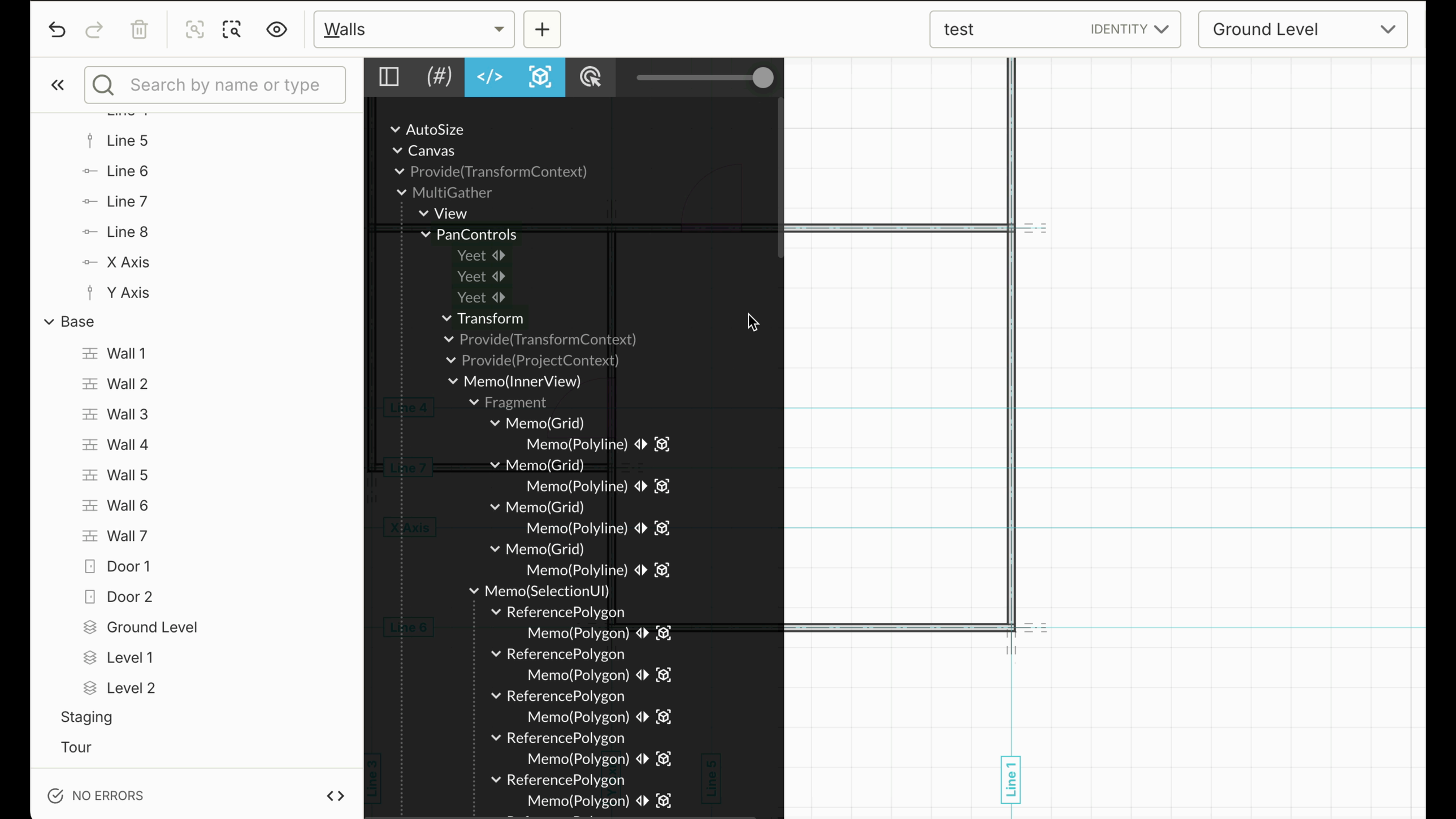
Y Axis

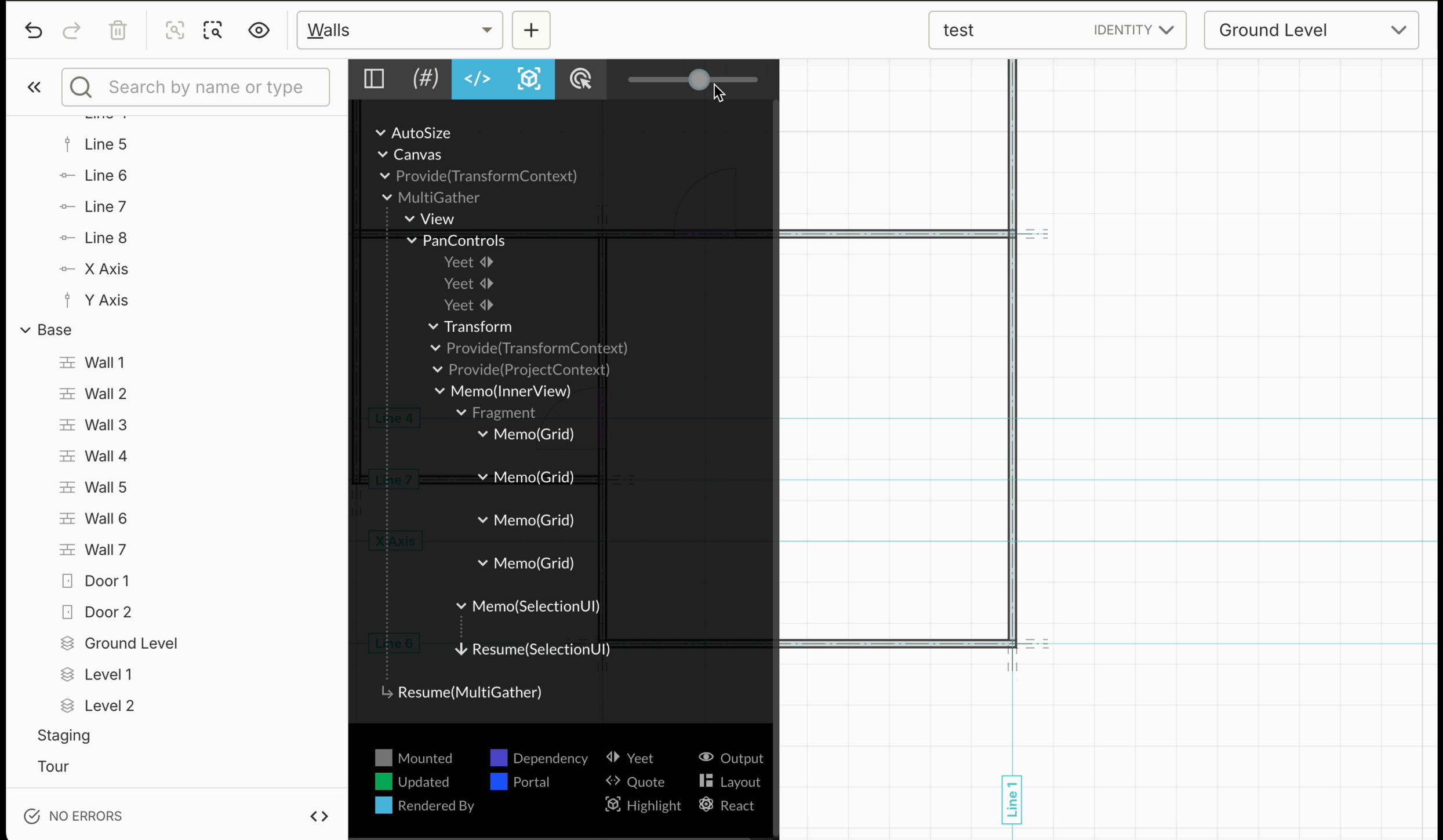
Line 5

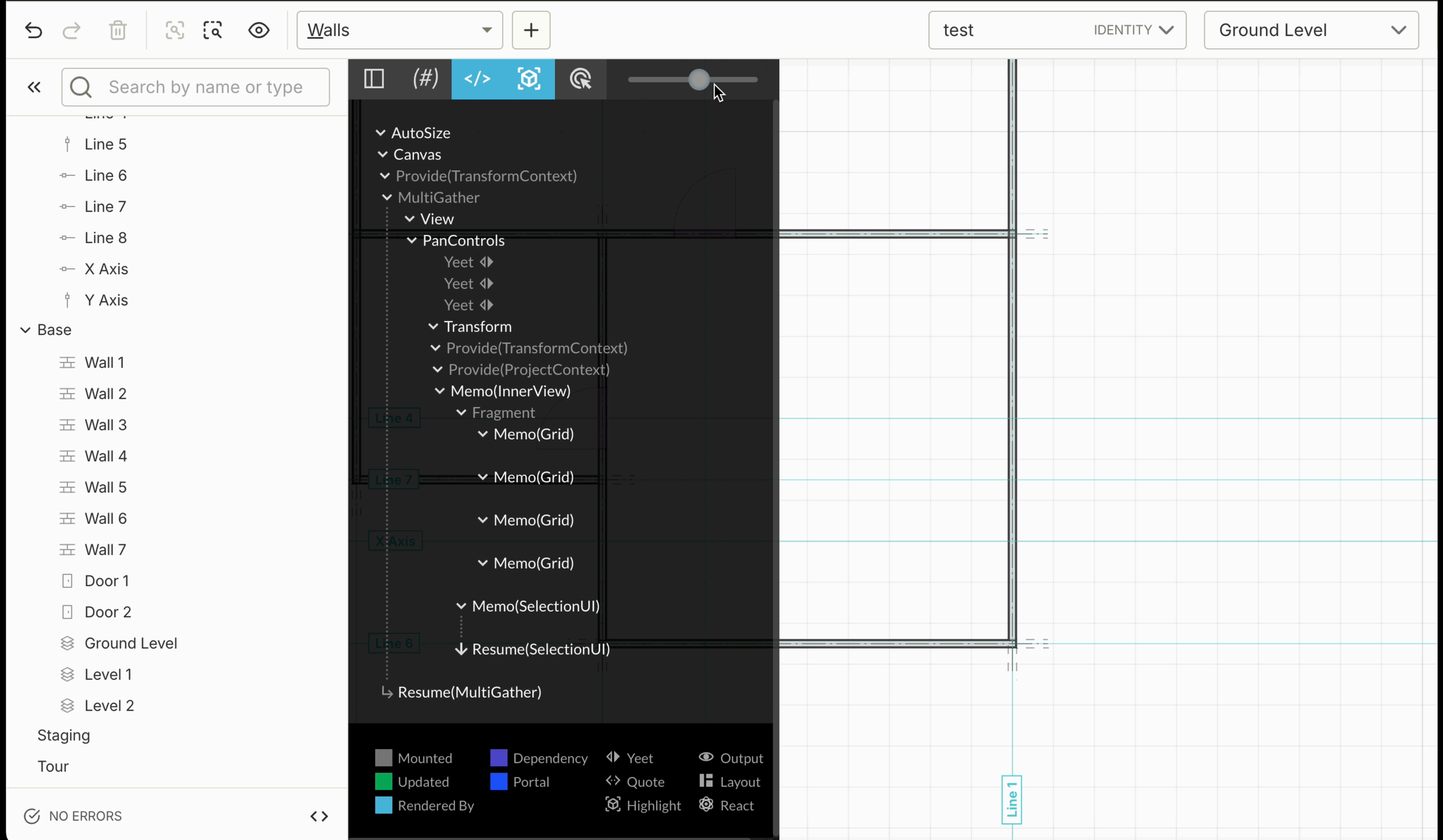

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
Line 4
Line 7
X Axis
Line 6
Line 3
Y Axis
Line 5
Line 1

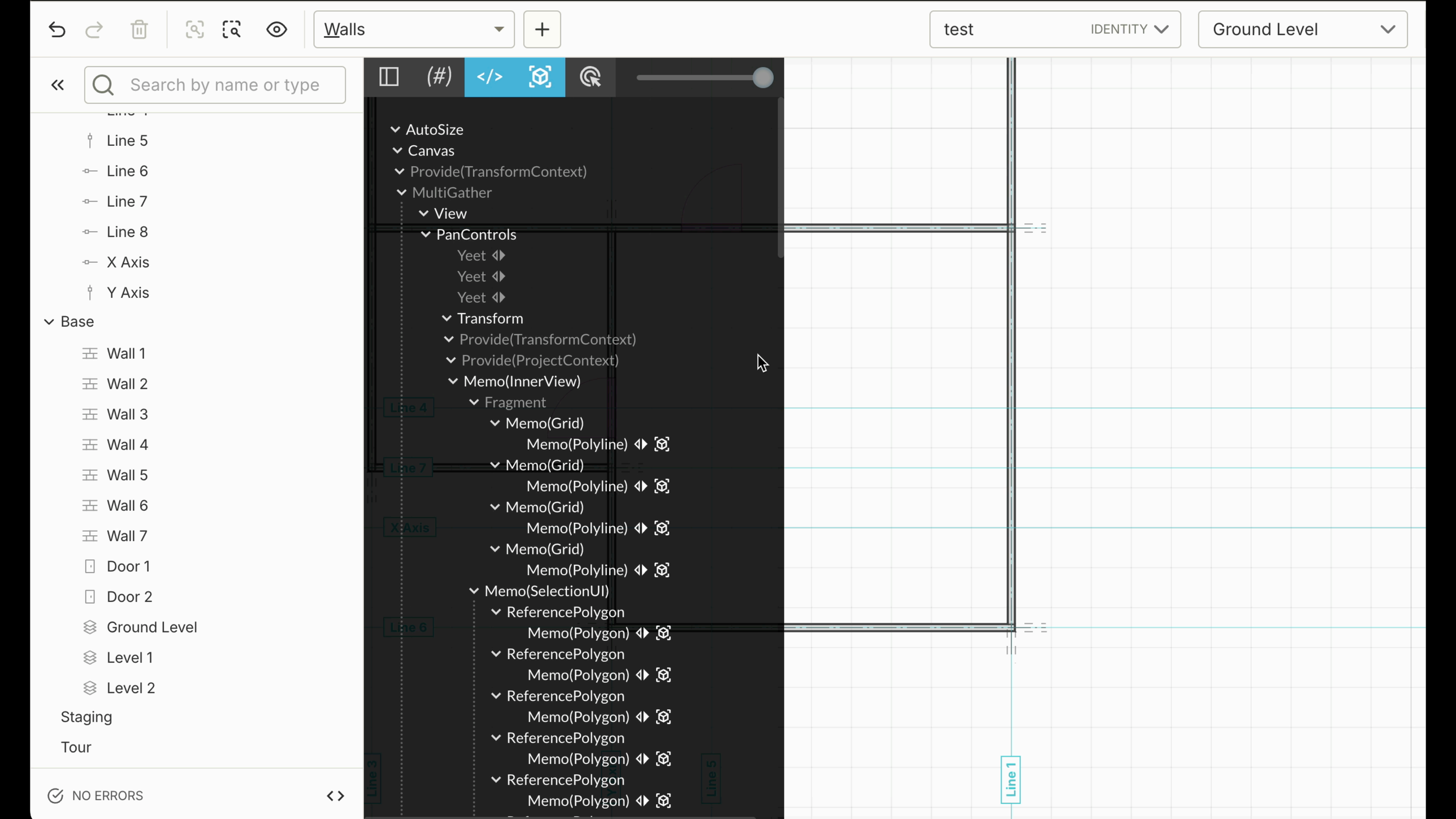

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
Line 4
Line 7
X Axis
Line 6
Line 3
Y Axis
Line 5
Line 1

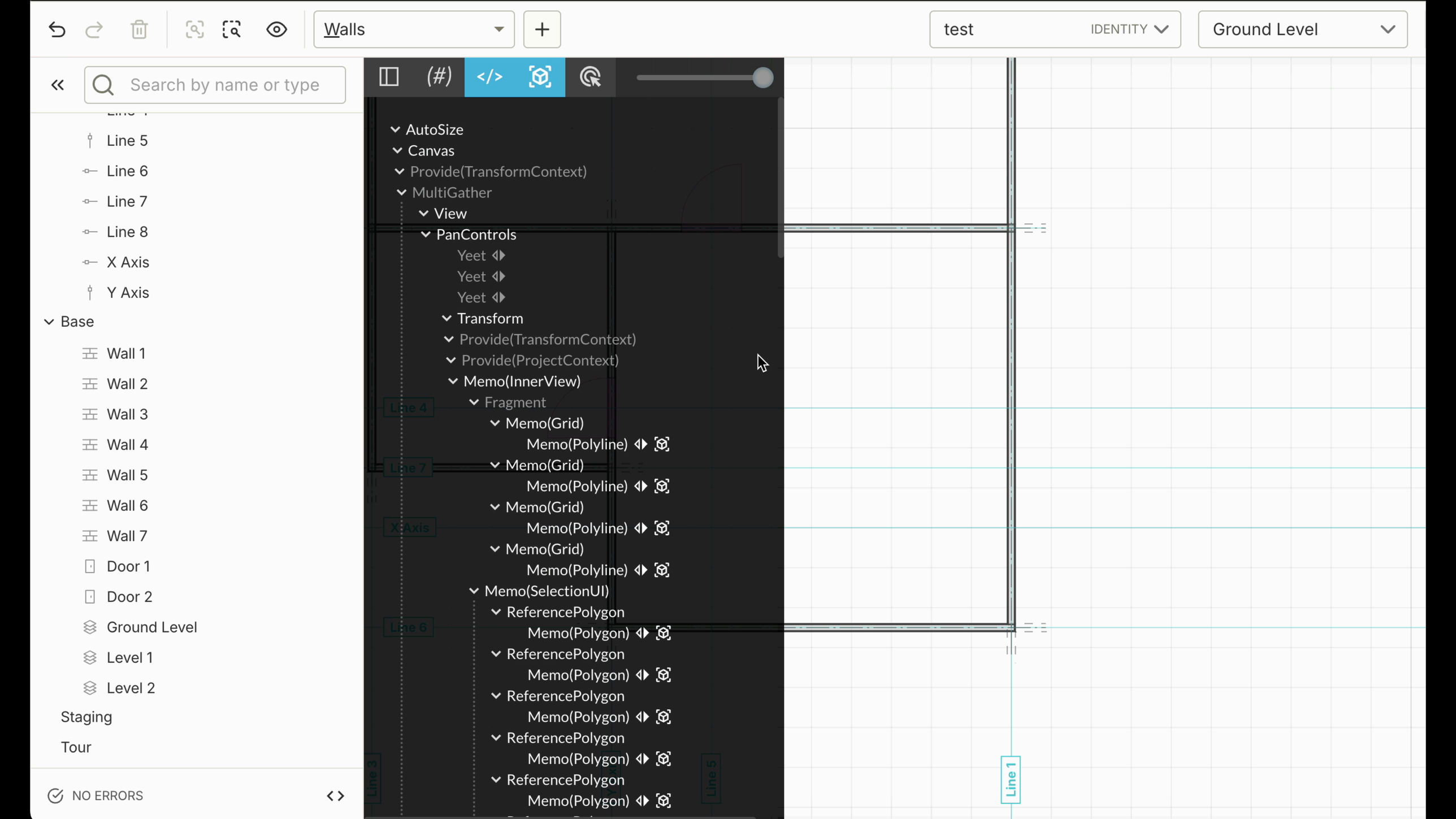

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
Line 4
Line 7
X Axis
Line 6
Line 3
Line 5
Line 1

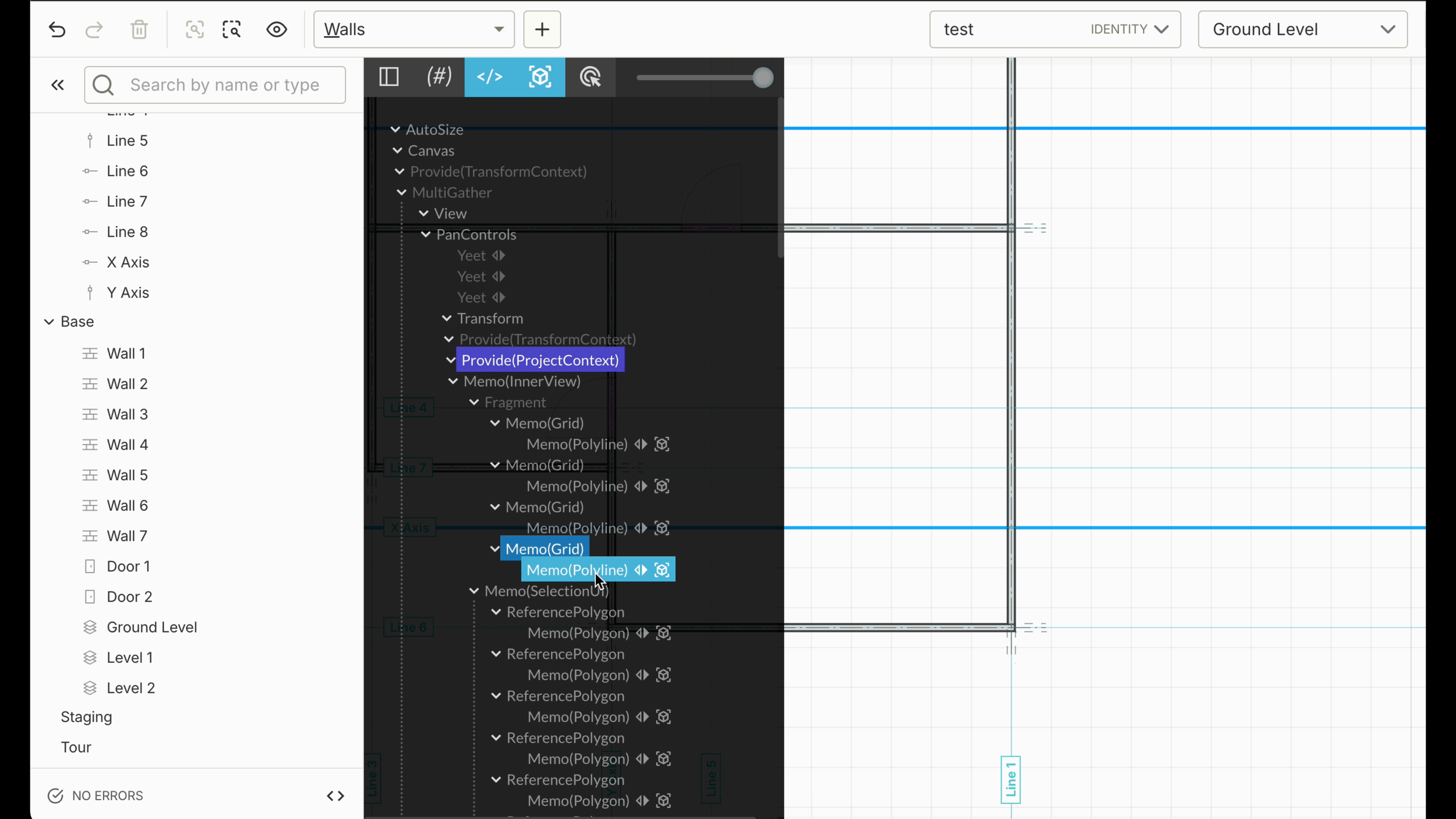

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
Line 4
Line 7
X Axis
Line 6
Line 3
Line 5
Line 1

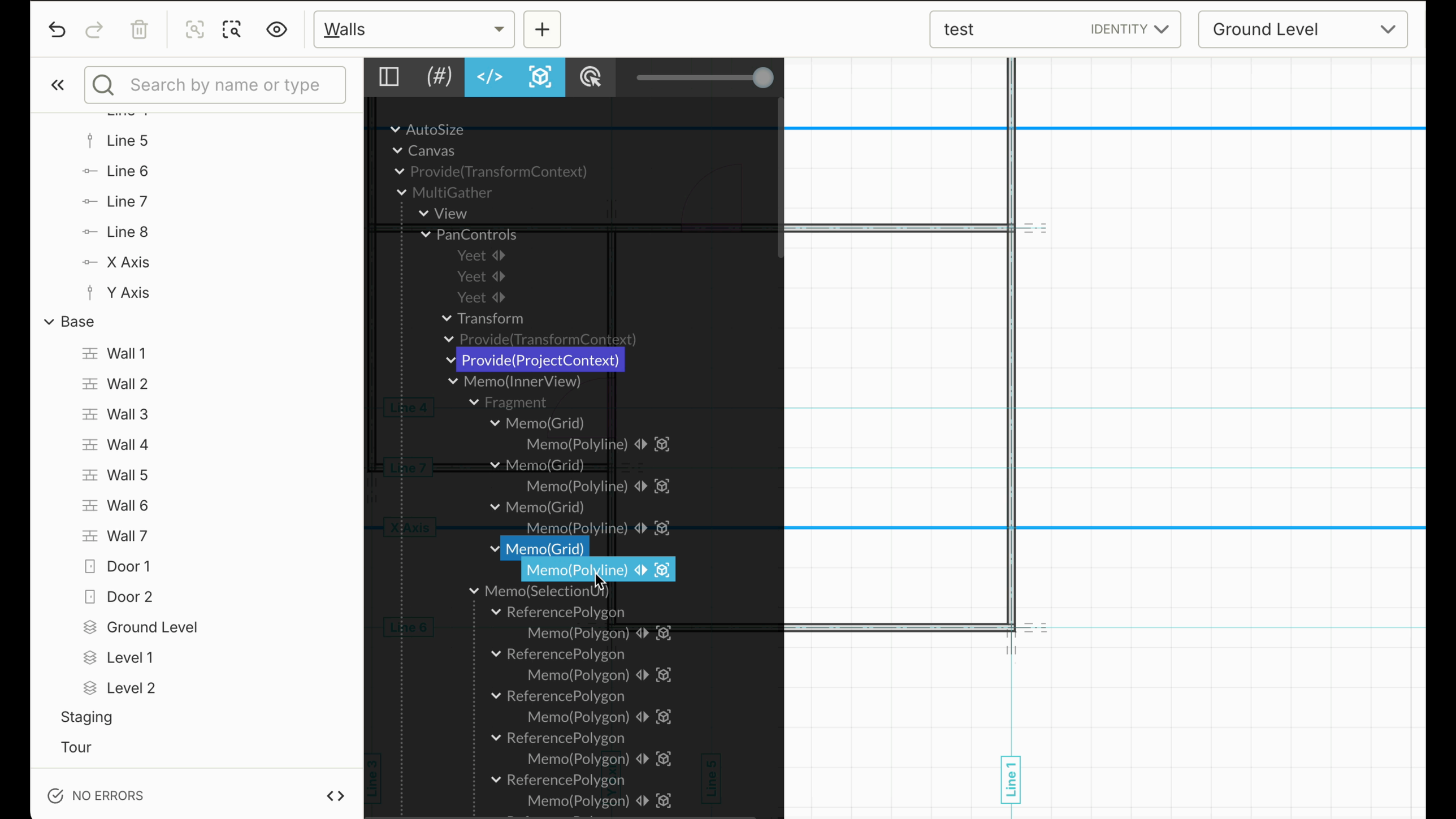

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(Grid)
Memo(Grid)
Memo(Grid)
Memo(Grid)
Memo(SelectionUI)
Resume(SelectionUI)
Resume(MultiGather)
Mounted
Updated
Rendered By
Dependency
Portal
Yeet
Quote
Highlight
Output
Layout
React
Line 1

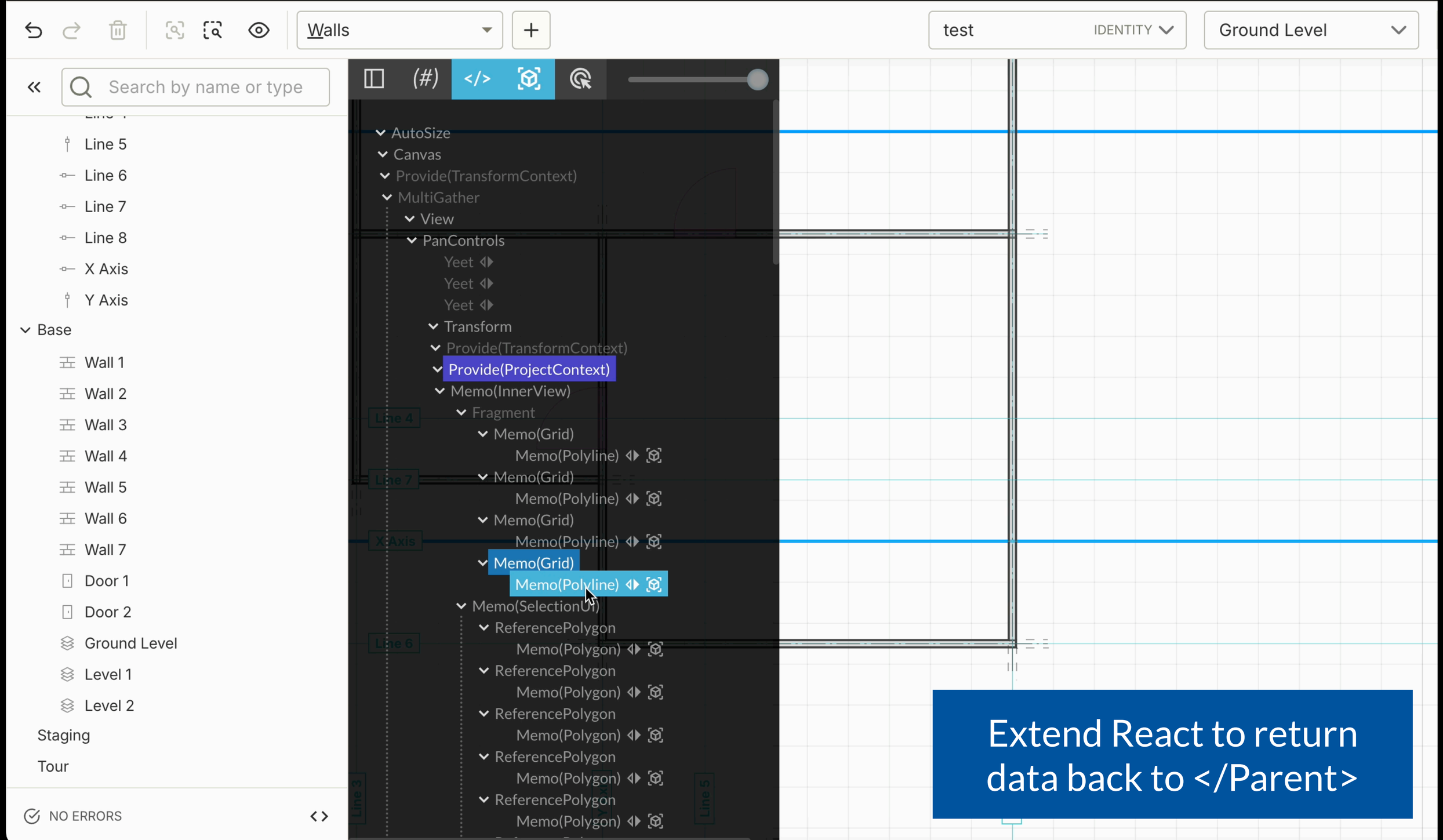

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(Grid)
Memo(Grid)
Memo(Grid)
Memo(Grid)
Memo(SelectionUI)
Resume(SelectionUI)
Resume(MultiGather)
Mounted
Updated
Rendered By
Dependency
Portal
Yeet
Quote
Highlight
Output
Layout
React
Line 1

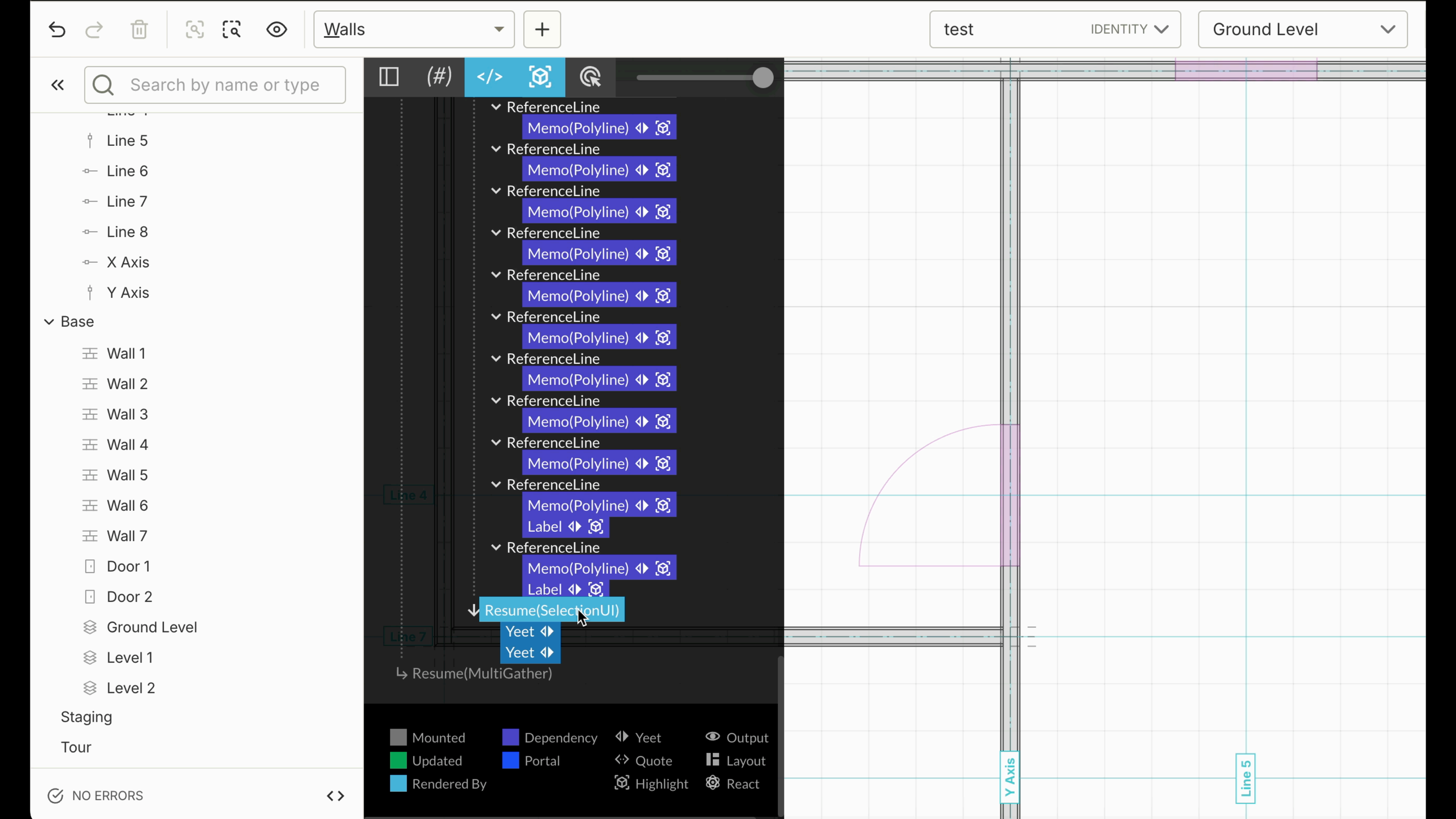

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
Line 4
Line 7
X Axis
Line 6
Line 3
Line 5
Line 1

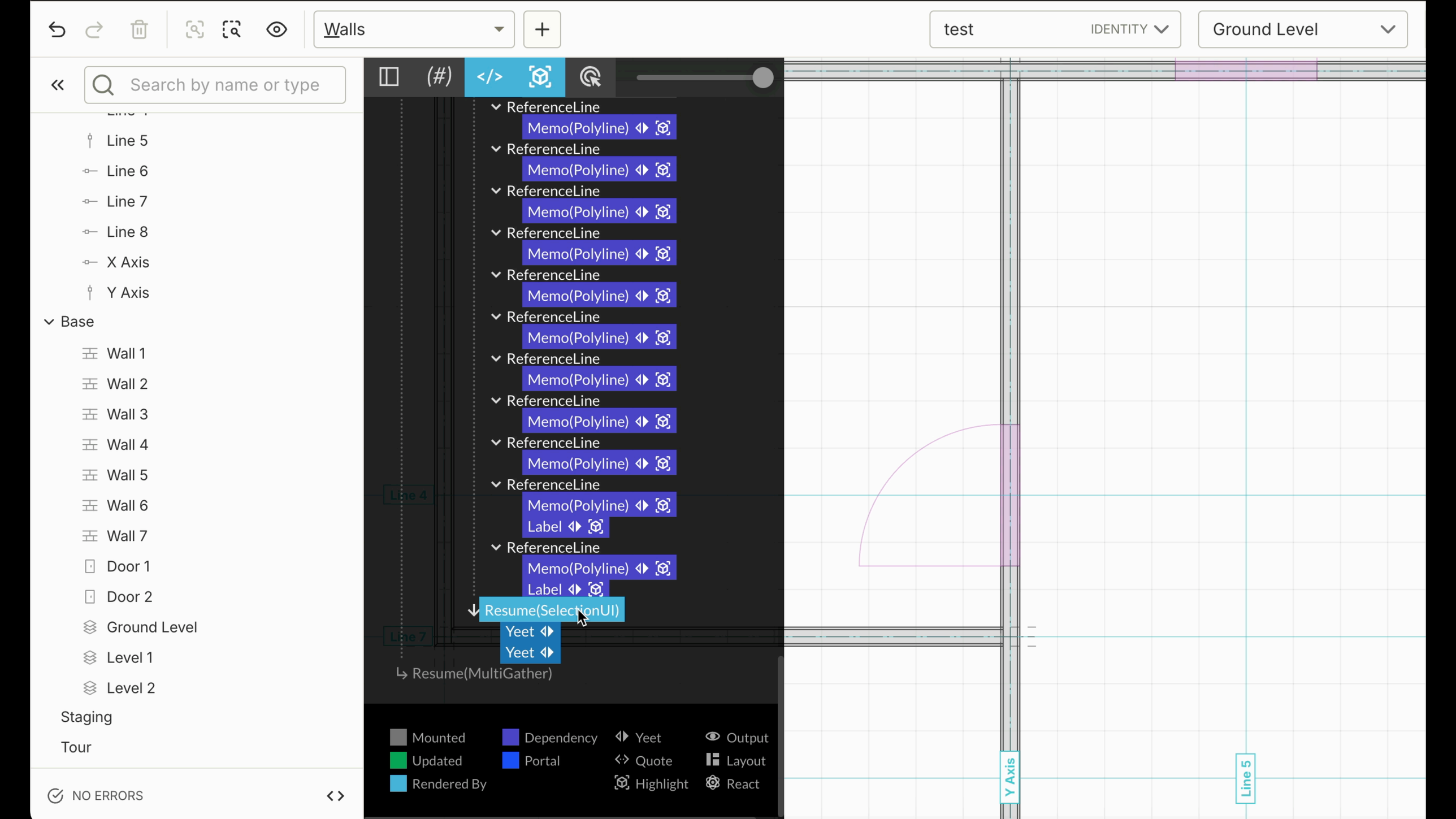

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
Line 4
Line 7
X Axis
Line 6
Line 3
Line 5
Line 1

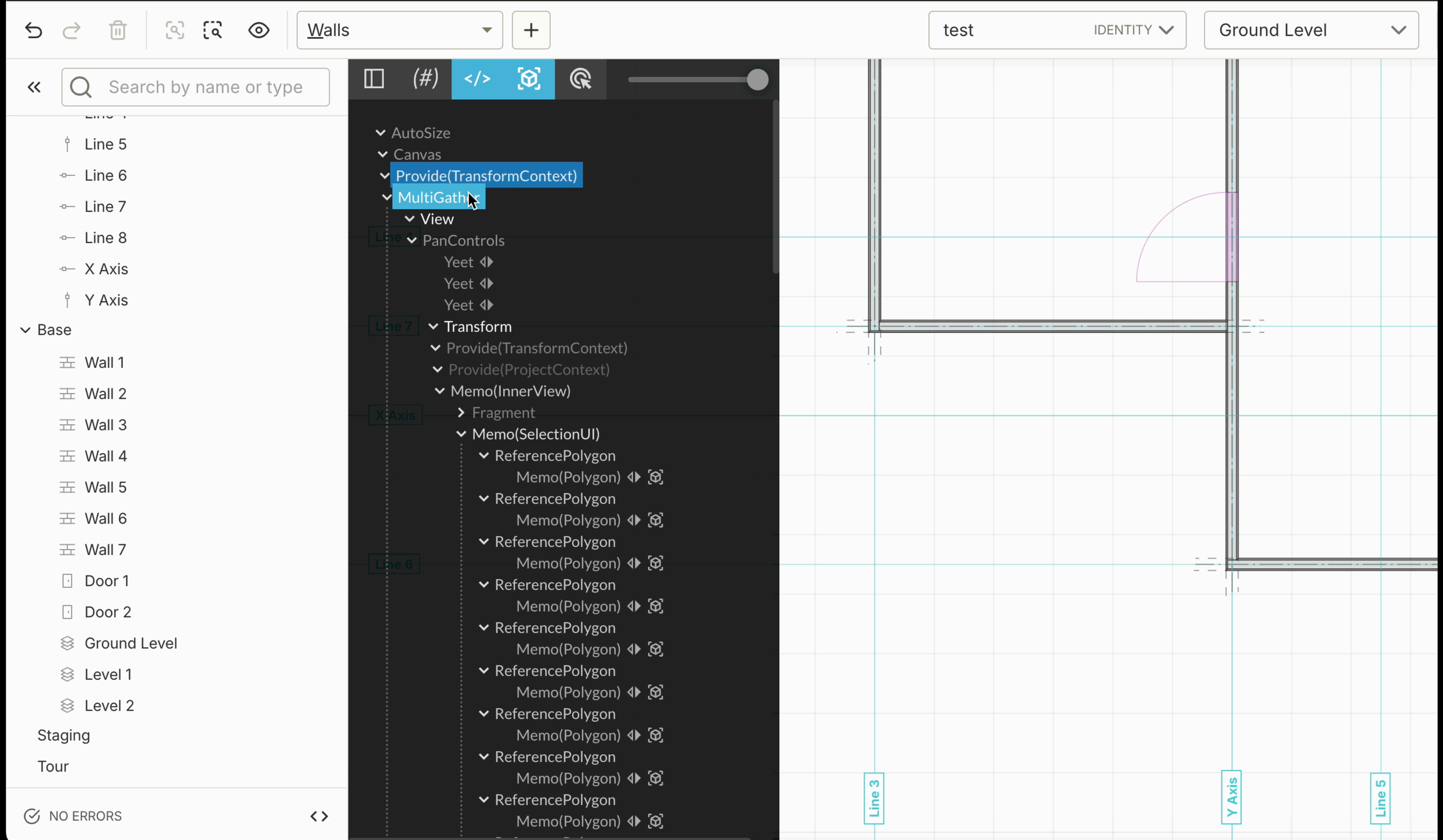

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
Line 4
Line 7
X Axis
Line 6
Line 3
Line 5
Line 1

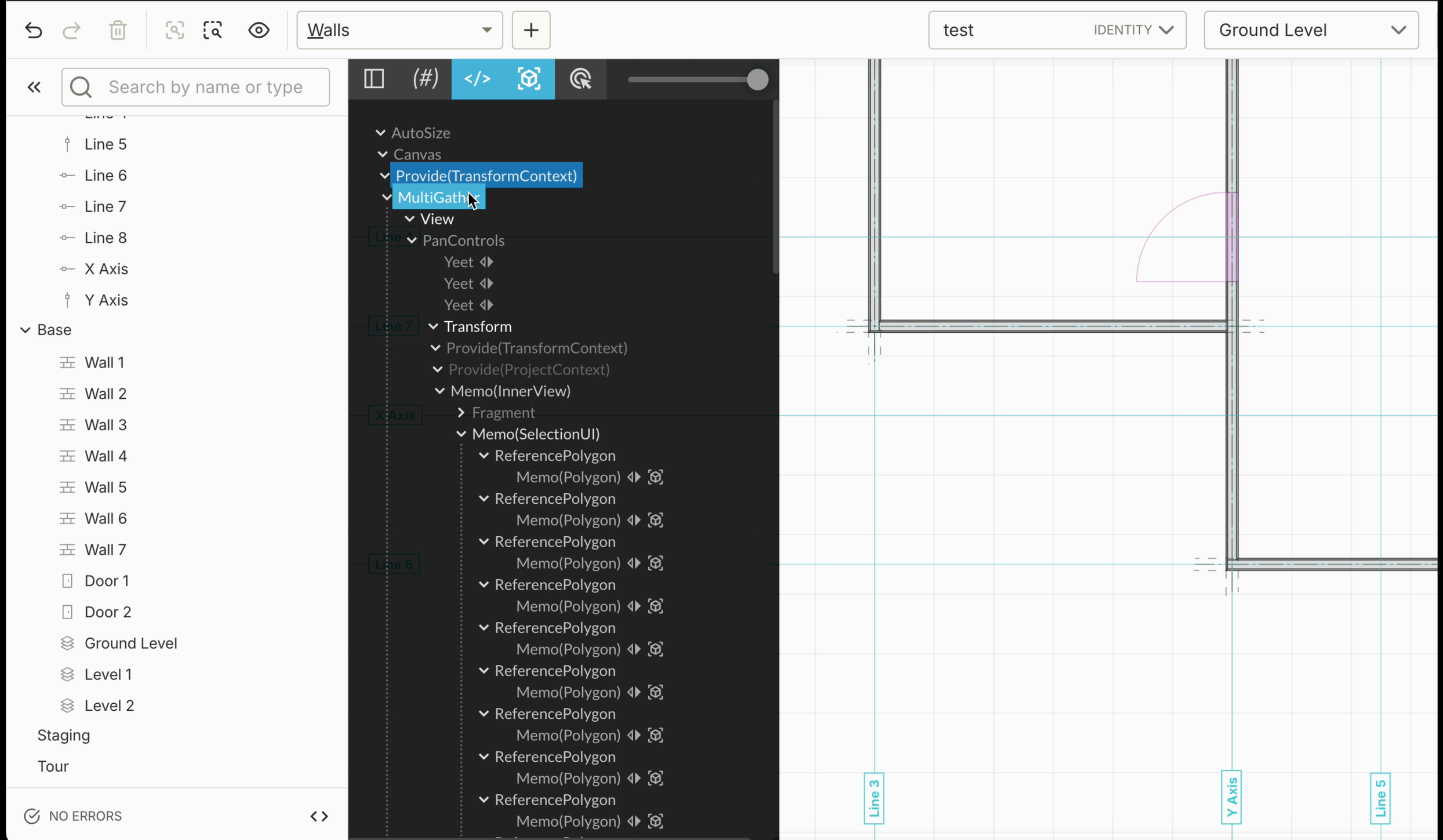

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
Line 4
Line 7
X Axis
Line 6
Line 3
Line 5
Line 1

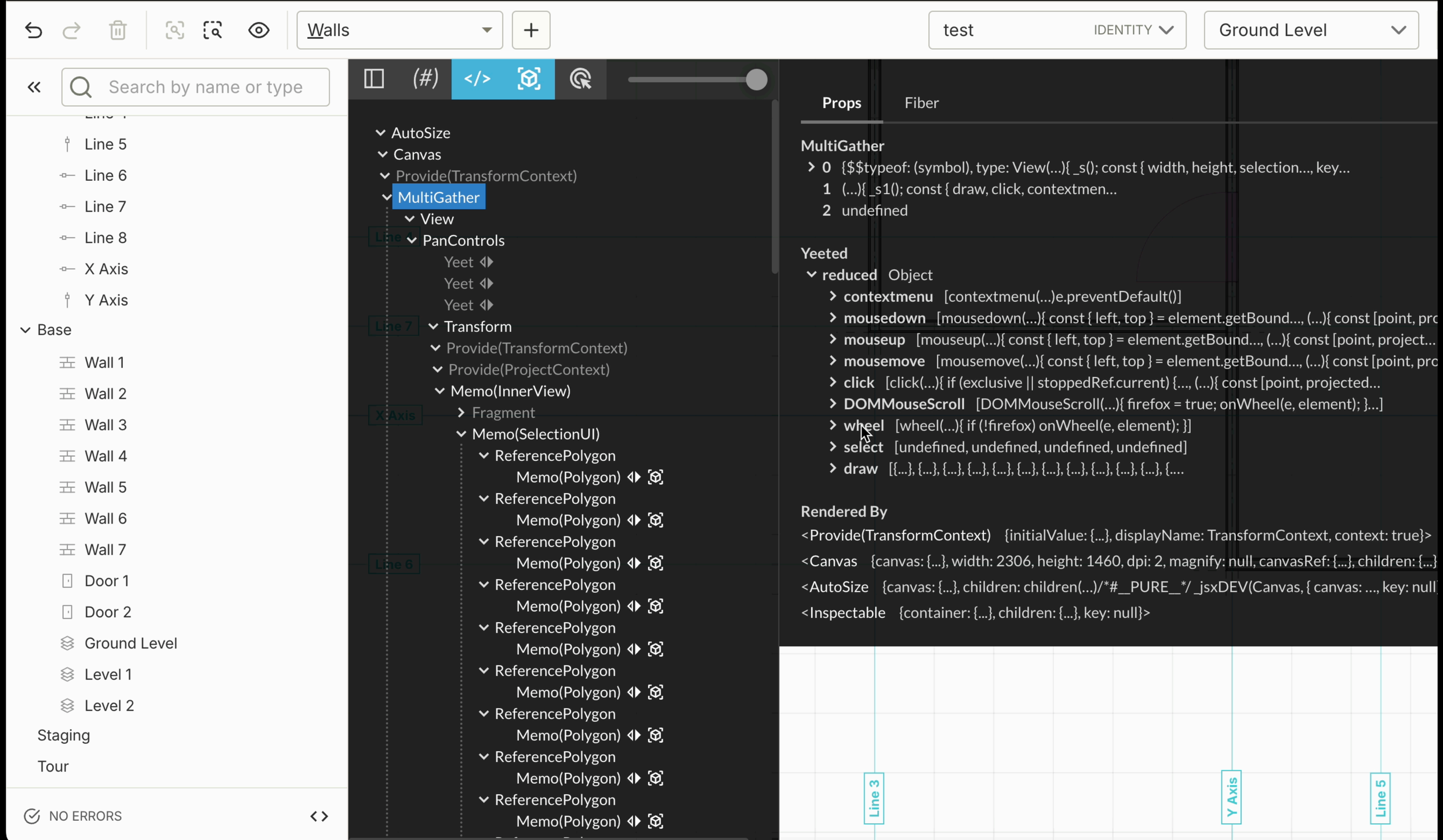

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(Grid)
Memo(Polyline)
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
ReferencePolygon
Memo(Polygon)
Line 4
Line 7
X Axis
Line 6
Line 3
Line 5
Extend React to return data back to </Parent>

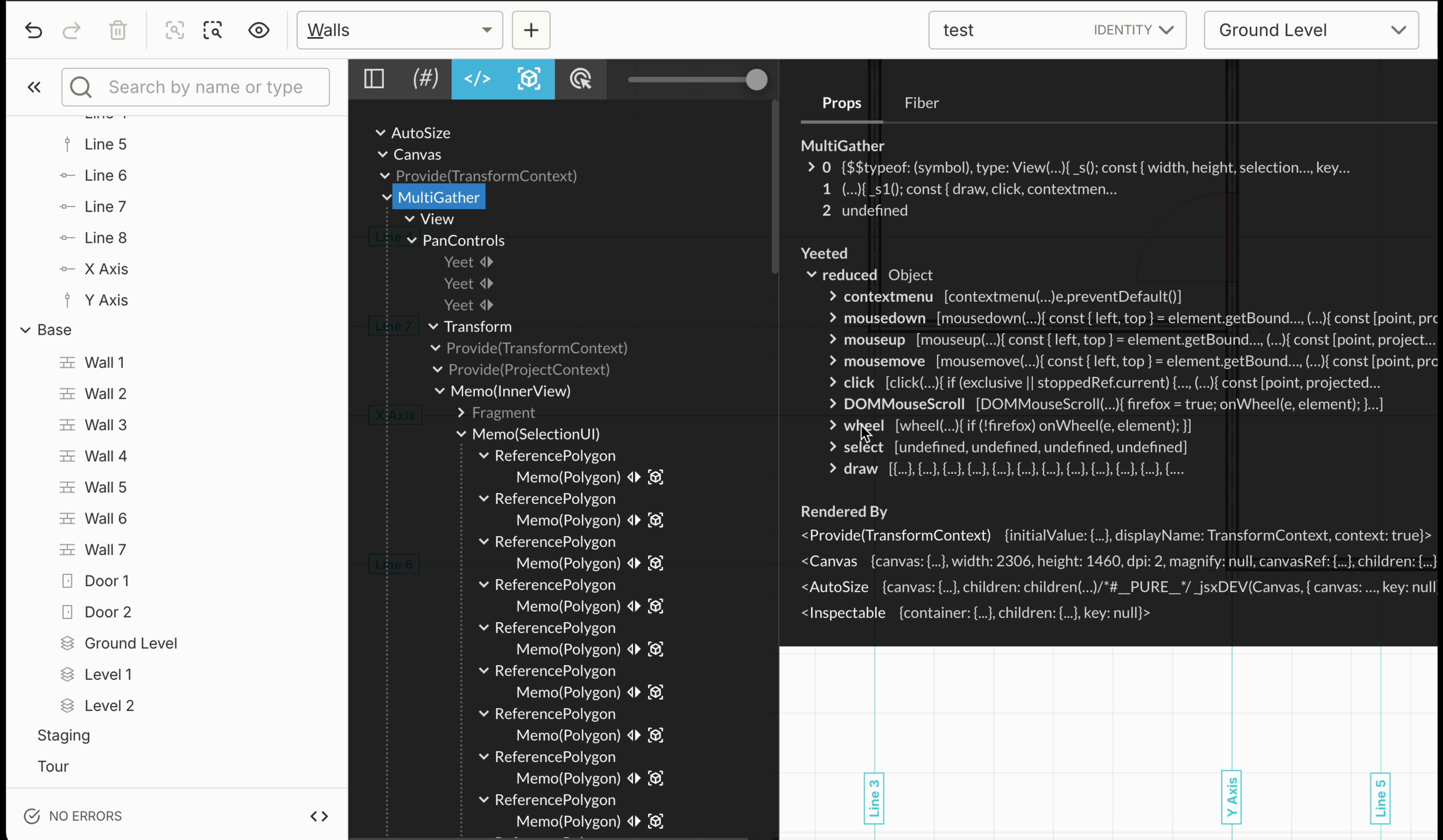

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
ReferenceLine
Memo(Polyline)
Label
Resume(SelectionUI)
Yeet
Resume(MultiGather)
Mounted
Dependency
Yeet
Output
Updated
Portal
Quote
Layout
Rendered By
Highlight
React
Line 4
Line 7
Y Axis
Line 5

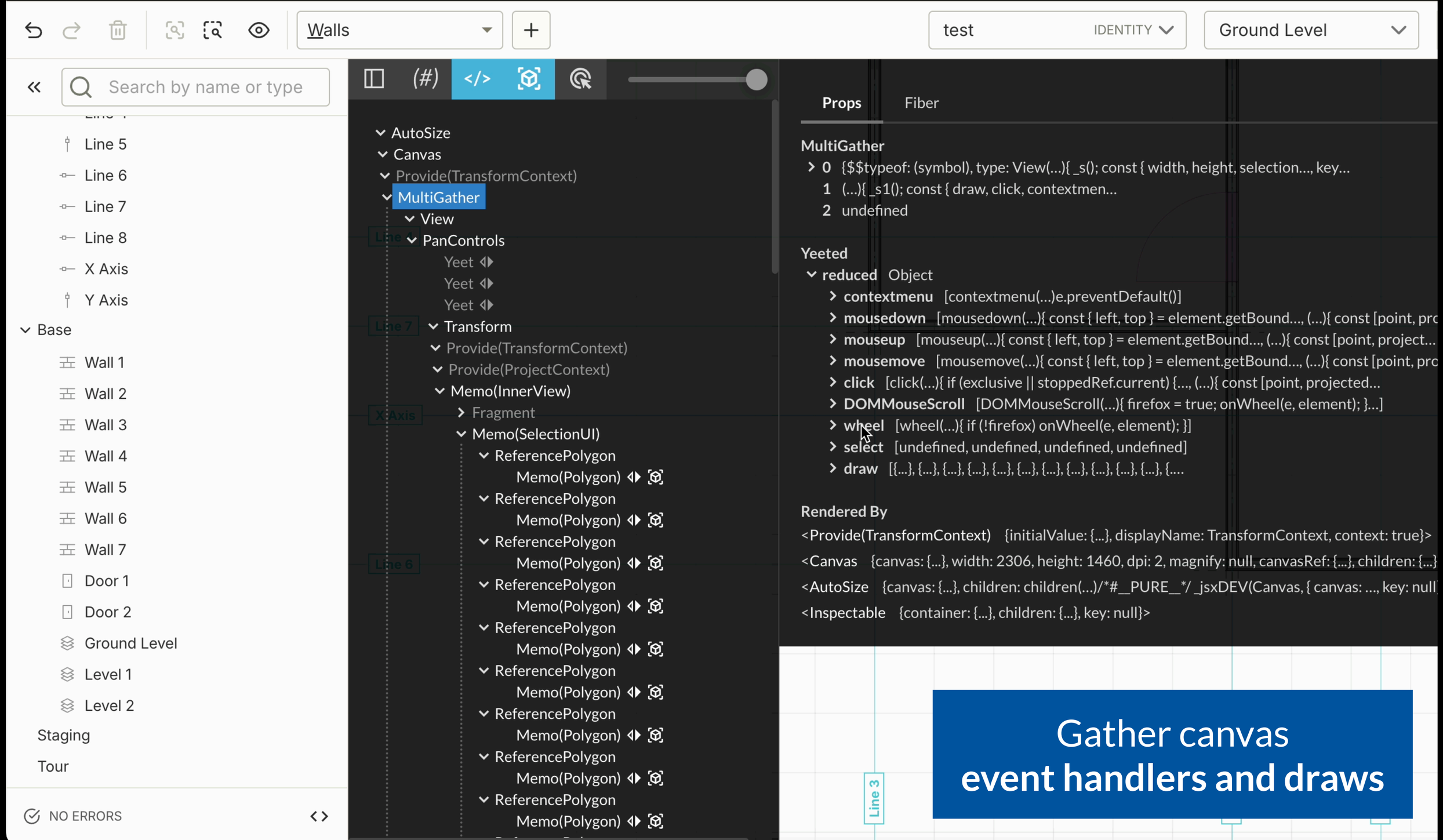

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
ReferenceLine
Memo(Polyline)
Label
Resume(SelectionUI)
Yeet
Resume(MultiGather)
Mounted
Dependency
Yeet
Output
Updated
Portal
Quote
Layout
Rendered By
Highlight
React
Line 4
Line 7
Y Axis
Line 5

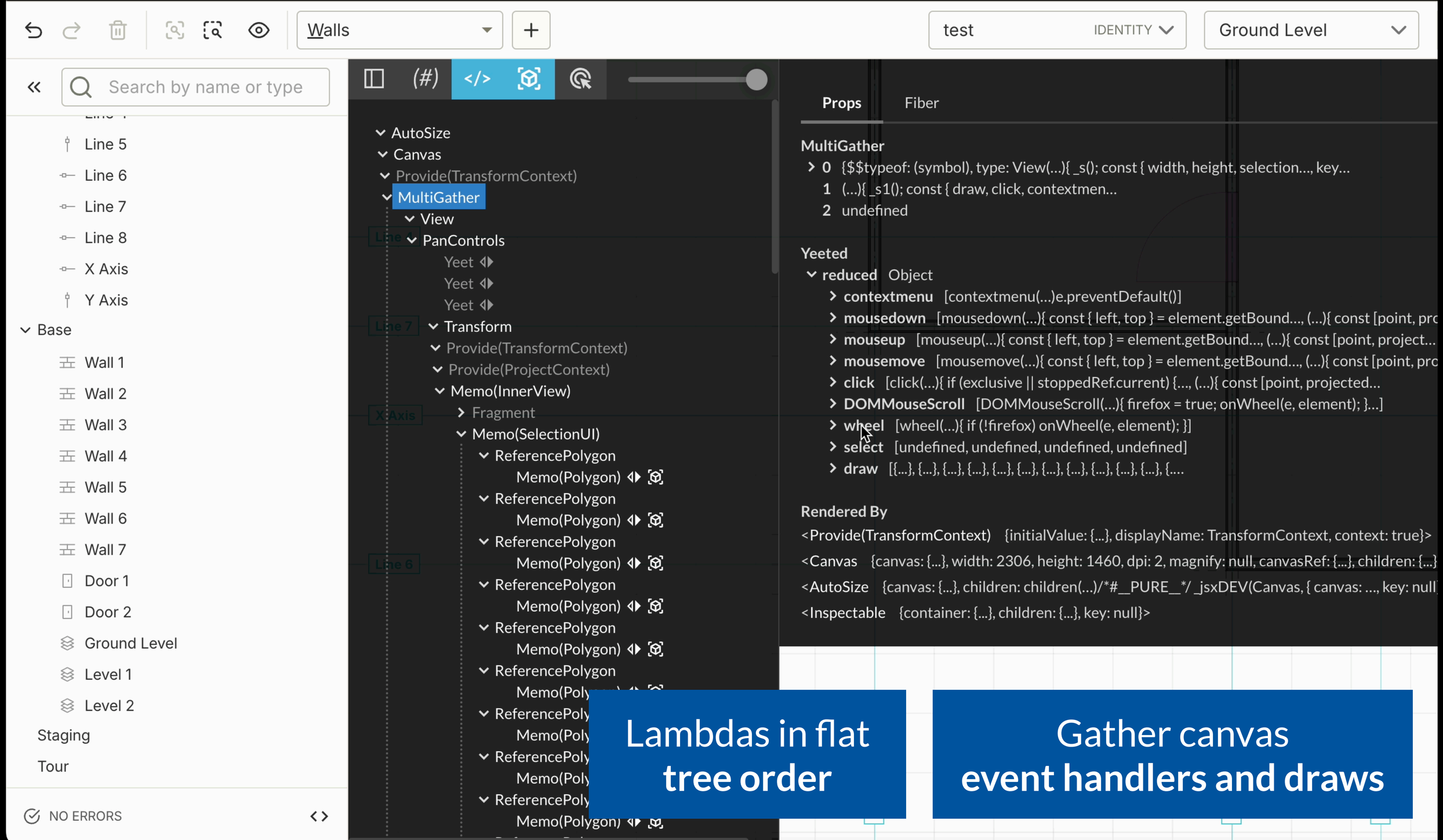

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGath
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
Line 3
Y Axis
Line 5

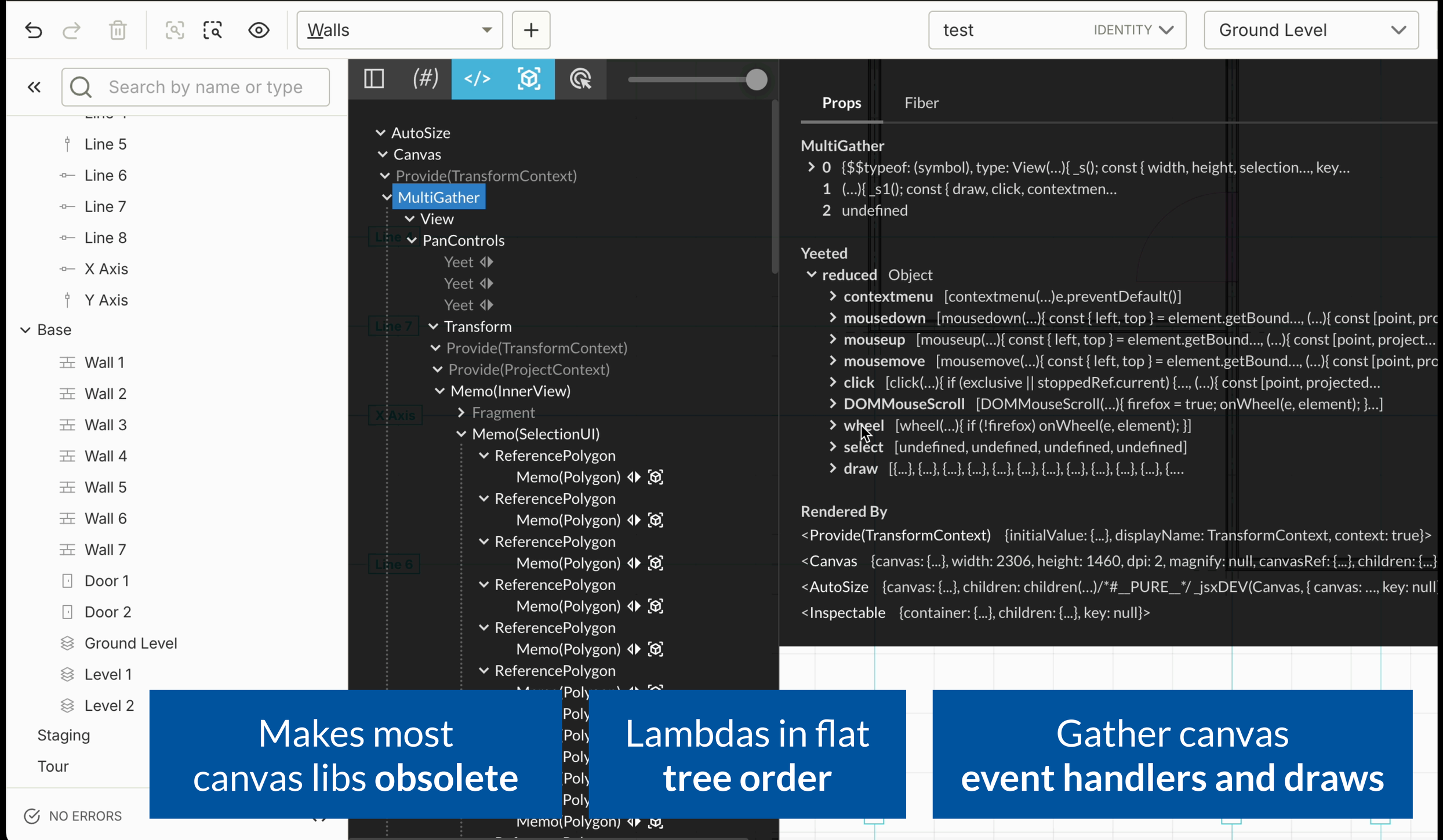

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGath
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
Line 3
Y Axis
Line 5

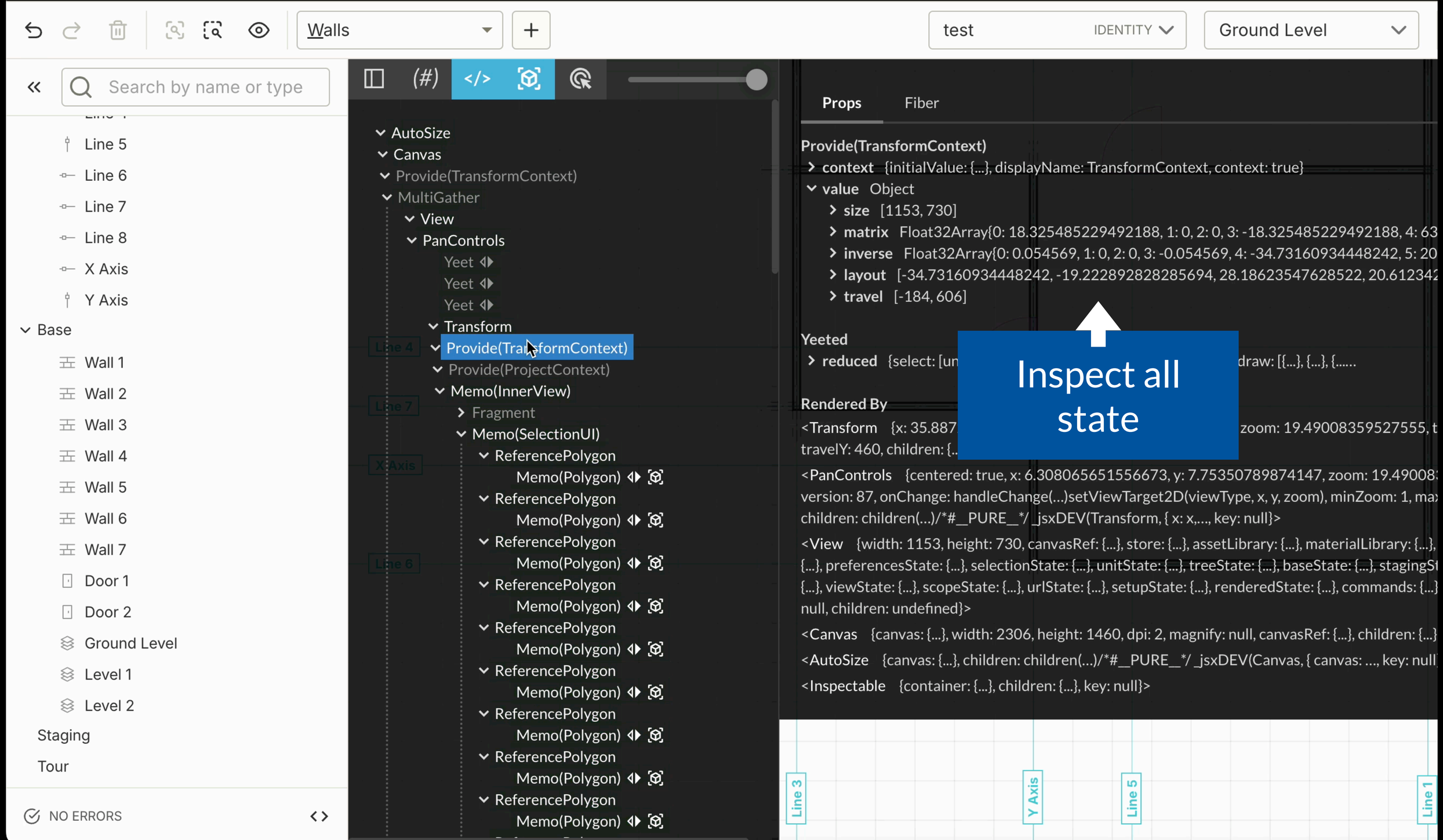

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
Props
Fiber
MultiGather
0 {$$typeof: (symbol), type: View(...){ _s(); const { width, height, selection..., key...
1 (...){ _s1(); const { draw, click, contextmen...
2 undefined
Yeeted
reduced Object
contextmenu [contextmenu(...)e.preventDefault()]
mousedown [mousedown(...){ const { left, top } = element.getBound..., (...){ const [point, pro
mouseup [mouseup(...){ const { left, top } = element.getBound..., (...){ const [point, project...
mousemove [mousemove(...){ const { left, top } = element.getBound..., (...){ const [point, pro
click [click(...){ if (exclusive || stoppedRef.current) {..., (...){ const [point, projected...
DOMMouseScroll [DOMMouseScroll(...){ firefox = true; onWheel(e, element); }...]
wheel [wheel(...){ if (!firefox) onWheel(e, element); }]
select [undefined, undefined, undefined, undefined]
draw [{...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {....
Rendered By
<Provide(TransformContext) {initialValue: {...}, displayName: TransformContext, context: true}>
<Canvas {canvas: {...}, width: 2306, height: 1460, dpi: 2, magnify: null, canvasRef: {...}, children: {...}
<AutoSize {canvas: {...}, children: children(...)/*#__PURE__*/ _jsxDEV(Canvas, { canvas: ..., key: null
<Inspectable {container: {...}, children: {...}, key: null}>
Line 3
Y Axis
Line 5

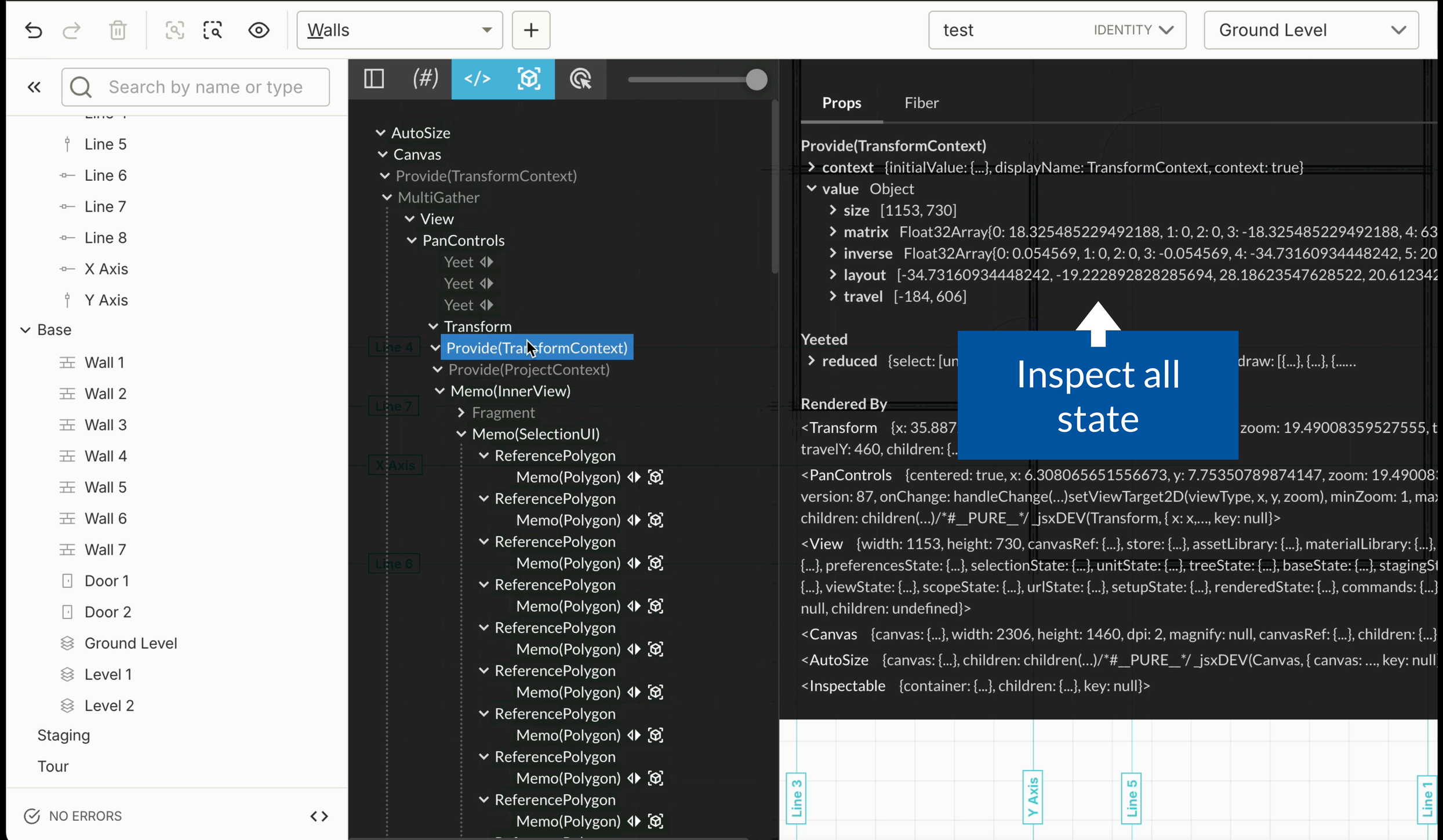

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
Props
Fiber
MultiGather
0 {$$typeof: (symbol), type: View(...){ _s(); const { width, height, selection..., key...
1 (...){ _s1(); const { draw, click, contextmen...
2 undefined
Yeeted
reduced Object
contextmenu [contextmenu(...)e.preventDefault()]
mousedown [mousedown(...){ const { left, top } = element.getBound..., (...){ const [point, pro
mouseup [mouseup(...){ const { left, top } = element.getBound..., (...){ const [point, project...
mousemove [mousemove(...){ const { left, top } = element.getBound..., (...){ const [point, pro
click [click(...){ if (exclusive || stoppedRef.current) {..., (...){ const [point, projected...
DOMMouseScroll [DOMMouseScroll(...){ firefox = true; onWheel(e, element); }...]
wheel [wheel(...){ if (!firefox) onWheel(e, element); }]
select [undefined, undefined, undefined, undefined]
draw [{...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {....
Rendered By
<Provide(TransformContext) {initialValue: {...}, displayName: TransformContext, context: true}>
<Canvas {canvas: {...}, width: 2306, height: 1460, dpi: 2, magnify: null, canvasRef: {...}, children: {...}
<AutoSize {canvas: {...}, children: children(...)/*#__PURE__*/ _jsxDEV(Canvas, { canvas: ..., key: null
<Inspectable {container: {...}, children: {...}, key: null}>
Line 3
Y Axis
Line 5

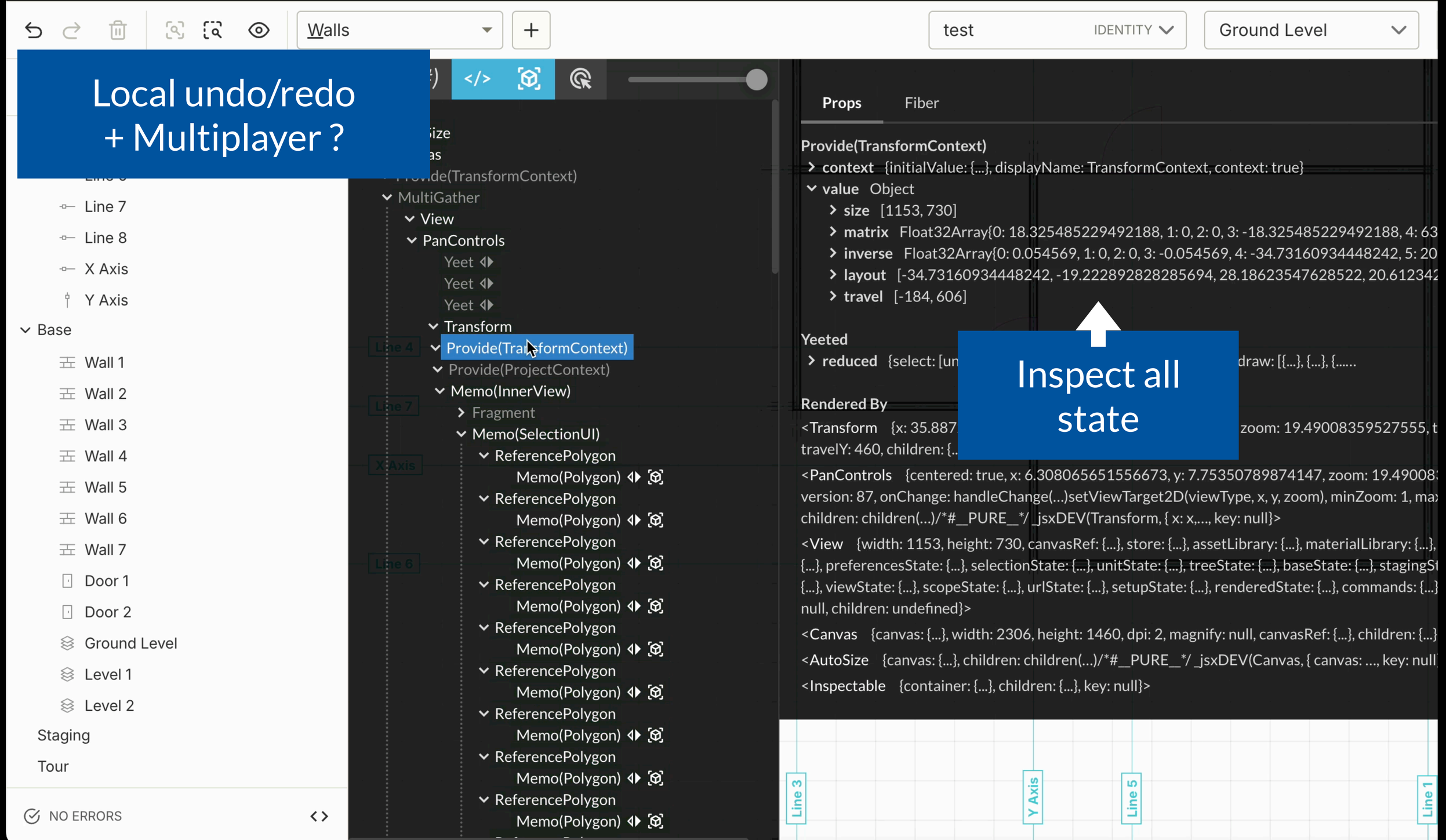

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
Props
Fiber
MultiGather
0 {$$typeof: (symbol), type: View(...){ _s(); const { width, height, selection..., key...
1 (...){ _s1(); const { draw, click, contextmen...
2 undefined
Yeeted
reduced Object
contextmenu [contextmenu(...)e.preventDefault()]
mousedown [mousedown(...){ const { left, top } = element.getBound..., (...){ const [point, pro
mouseup [mouseup(...){ const { left, top } = element.getBound..., (...){ const [point, project...
mousemove [mousemove(...){ const { left, top } = element.getBound..., (...){ const [point, pro
click [click(...){ if (exclusive || stoppedRef.current) {..., (...){ const [point, projected...
DOMMouseScroll [DOMMouseScroll(...){ firefox = true; onWheel(e, element); }...]
wheel [wheel(...){ if (!firefox) onWheel(e, element); }]
select [undefined, undefined, undefined, undefined]
draw [{...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {....
Rendered By
<Provide(TransformContext) {initialValue: {...}, displayName: TransformContext, context: true}>
<Canvas {canvas: {...}, width: 2306, height: 1460, dpi: 2, magnify: null, canvasRef: {...}, children: {...}
<AutoSize {canvas: {...}, children: children(...)/*#__PURE__*/ _jsxDEV(Canvas, { canvas: ..., key: null
<Inspectable {container: {...}, children: {...}, key: null}>
Line 3
Gather canvas
event handlers and draws

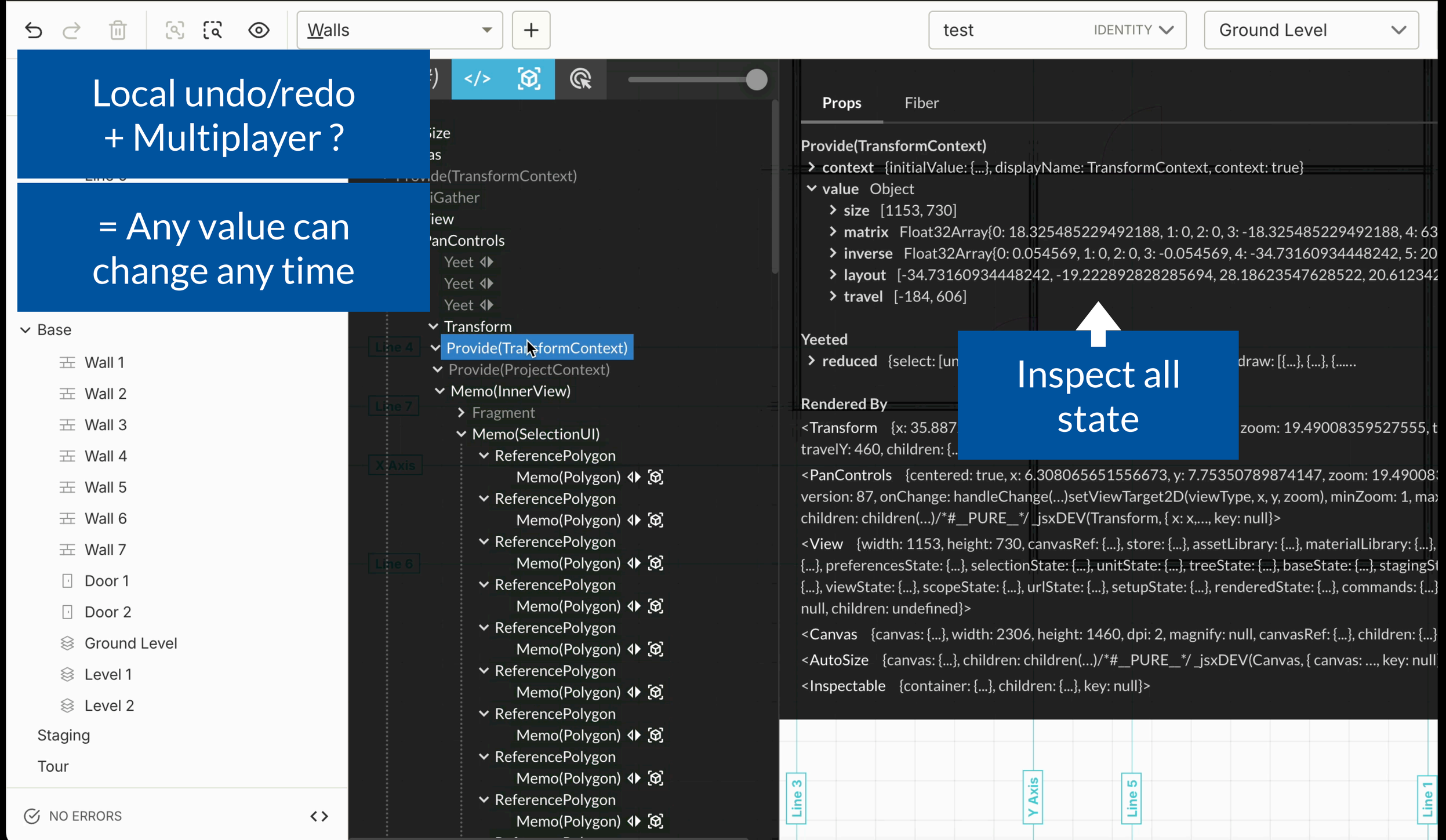

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
Props
Fiber
MultiGather
0 {$$typeof: (symbol), type: View(...){ _s(); const { width, height, selection..., key...
1 (...){ _s1(); const { draw, click, contextmen...
2 undefined
Yeeted
reduced Object
contextmenu [contextmenu(...)e.preventDefault()]
mousedown [mousedown(...){ const { left, top } = element.getBound..., (...){ const [point, pro
mouseup [mouseup(...){ const { left, top } = element.getBound..., (...){ const [point, project...
mousemove [mousemove(...){ const { left, top } = element.getBound..., (...){ const [point, pro
click [click(...){ if (exclusive || stoppedRef.current) {..., (...){ const [point, projected...
DOMMouseScroll [DOMMouseScroll(...){ firefox = true; onWheel(e, element); }...]
wheel [wheel(...){ if (!firefox) onWheel(e, element); }]
select [undefined, undefined, undefined, undefined]
draw [{...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {....
Rendered By
<Provide(TransformContext) {initialValue: {...}, displayName: TransformContext, context: true}>
<Canvas {canvas: {...}, width: 2306, height: 1460, dpi: 2, magnify: null, canvasRef: {...}, children: {...}
<AutoSize {canvas: {...}, children: children(...)/*#__PURE__*/ _jsxDEV(Canvas, { canvas: ..., key: null
<Inspectable {container: {...}, children: {...}, key: null}>
Lambdas in flat tree order
Gather canvas event handlers and draws

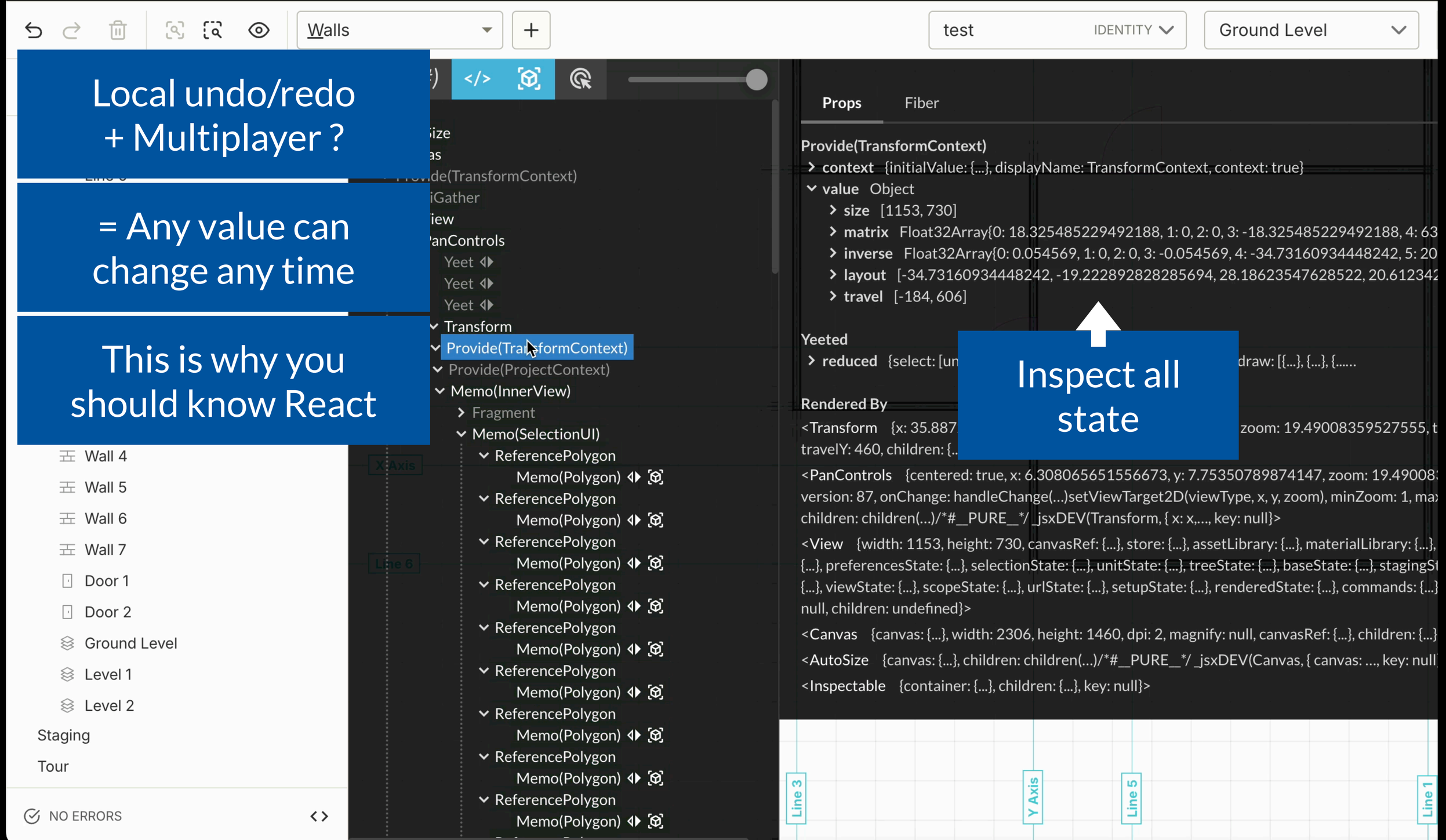

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
Props
Fiber
MultiGather
0 {$$typeof: (symbol), type: View(...){ _s(); const { width, height, selection..., key...
1 (...){ _s1(); const { draw, click, contextmen...
2 undefined
Yeeted
reduced Object
contextmenu [contextmenu(...)e.preventDefault()]
mousedown [mousedown(...){ const { left, top } = element.getBound..., (...){ const [point, pro
mouseup [mouseup(...){ const { left, top } = element.getBound..., (...){ const [point, project...
mousemove [mousemove(...){ const { left, top } = element.getBound..., (...){ const [point, pro
click [click(...){ if (exclusive || stoppedRef.current) {..., (...){ const [point, projected...
DOMMouseScroll [DOMMouseScroll(...){ firefox = true; onWheel(e, element); }...]
wheel [wheel(...){ if (!firefox) onWheel(e, element); }]
select [undefined, undefined, undefined, undefined]
draw [{...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {....
Rendered By
<Provide(TransformContext) {initialValue: {...}, displayName: TransformContext, context: true}>
<Canvas {canvas: {...}, width: 2306, height: 1460, dpi: 2, magnify: null, canvasRef: {...}, children: {...}
<AutoSize {canvas: {...}, children: children(...)/*#__PURE__*/ _jsxDEV(Canvas, { canvas: ..., key: null
<Inspectable {container: {...}, children: {...}, key: null}>
Makes most canvas libs obsolete
Lambdas in flat tree order
Gather canvas event handlers and draws

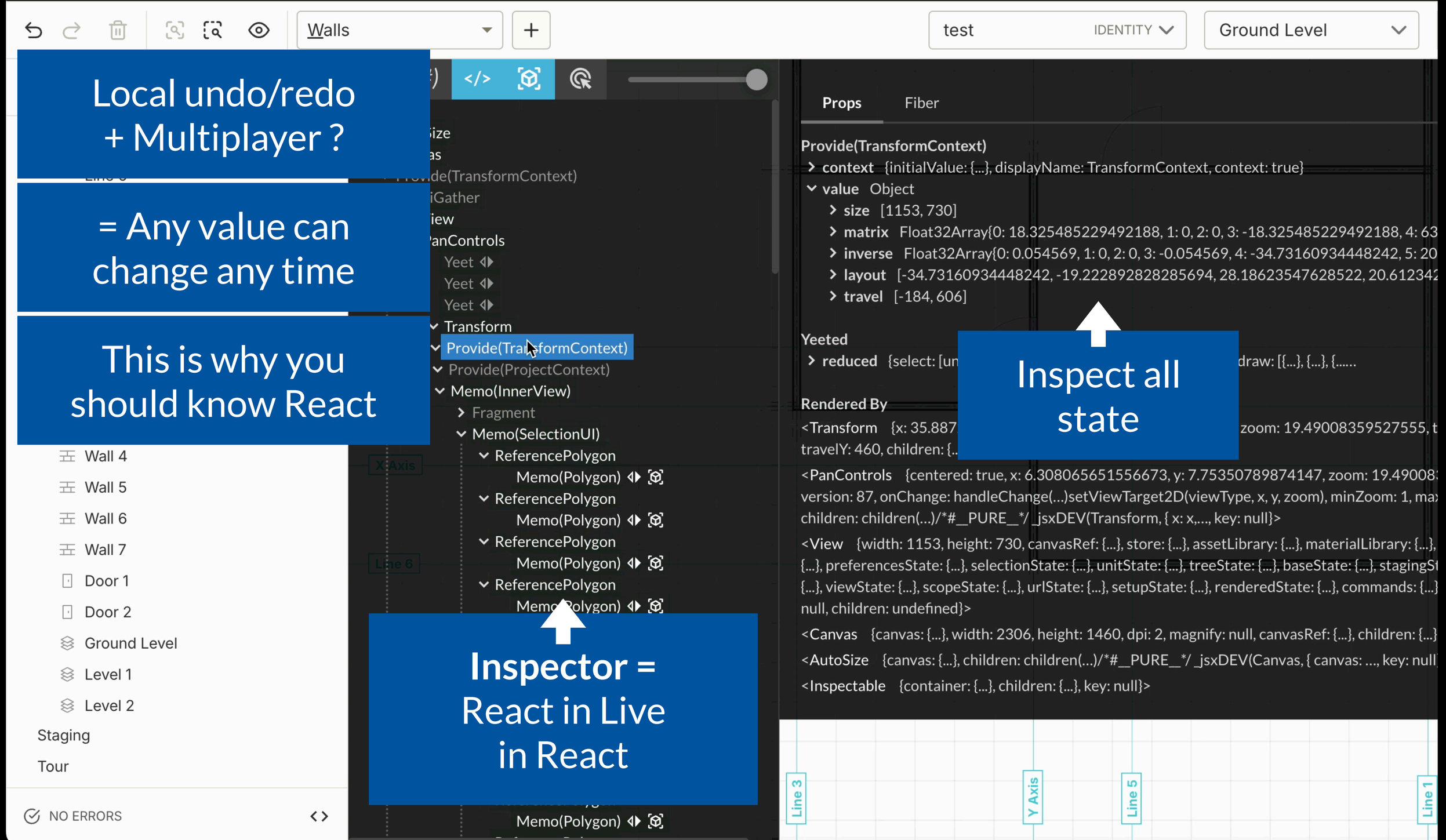

Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
Props
Fiber
Provide(TransformContext)
context {initialValue: {...}, displayName: TransformContext, context: true}
value Object
size [1153, 730]
matrix Float32Array{0: 18.325485229492188, 1: 0, 2: 0, 3: -18.325485229492188, 4: 63
inverse Float32Array{0: 0.054569, 1: 0, 2: 0, 3: -0.054569, 4: -34.73160934448242, 5: 20
layout [-34.73160934448242, -19.222892828285694, 28.18623547628522, 20.612342
travel [-184, 606]
Yeeted
reduced {select: [un
draw: [{...}, {...}, {.....
Rendered By
<Transform {x: 35.887
zoom: 19.49008359527555, t
travelY: 460, children: {..
<PanControls {centered: true, x: 6.308065651556673, y: 7.75350789874147, zoom: 19.49008
version: 87, onChange: handleChange(...)setViewTarget2D(viewType, x, y, zoom), minZoom: 1, ma
children: children(...)/*#__PURE__*/ _jsxDEV(Transform, { x: x,..., key: null}>
<View {width: 1153, height: 730, canvasRef: {...}, store: {...}, assetLibrary: {...}, materialLibrary: {...},
{...}, preferencesState: {...}, selectionState: {...}, unitState: {...}, treeState: {...}, baseState: {...}, stagingSt
{...}, viewState: {...}, scopeState: {...}, urlState: {...}, setupState: {...}, renderedState: {...}, commands: {...}
null, children: undefined}>
<Canvas {canvas: {...}, width: 2306, height: 1460, dpi: 2, magnify: null, canvasRef: {...}, children: {...}
<AutoSize {canvas: {...}, children: children(...)/*#__PURE__*/ _jsxDEV(Canvas, { canvas: ..., key: null
<Inspectable {container: {...}, children: {...}, key: null}>
Inspect all state
Line 3
Y Axis
Line 5
Line 1


Walls
test
IDENTITY
Ground Level
Search by name or type
Line 5
Line 6
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
AutoSize
Canvas
Provide(TransformContext)
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
Props
Fiber
Provide(TransformContext)
context {initialValue: {...}, displayName: TransformContext, context: true}
value Object
size [1153, 730]
matrix Float32Array{0: 18.325485229492188, 1: 0, 2: 0, 3: -18.325485229492188, 4: 63
inverse Float32Array{0: 0.054569, 1: 0, 2: 0, 3: -0.054569, 4: -34.73160934448242, 5: 20
layout [-34.73160934448242, -19.222892828285694, 28.18623547628522, 20.612342
travel [-184, 606]
Yeeted
reduced {select: [un
draw: [{...}, {...}, {.....
Rendered By
<Transform {x: 35.887
zoom: 19.49008359527555, t
travelY: 460, children: {..
<PanControls {centered: true, x: 6.308065651556673, y: 7.75350789874147, zoom: 19.49008
version: 87, onChange: handleChange(...)setViewTarget2D(viewType, x, y, zoom), minZoom: 1, ma
children: children(...)/*#__PURE__*/ _jsxDEV(Transform, { x: x,..., key: null}>
<View {width: 1153, height: 730, canvasRef: {...}, store: {...}, assetLibrary: {...}, materialLibrary: {...},
{...}, preferencesState: {...}, selectionState: {...}, unitState: {...}, treeState: {...}, baseState: {...}, stagingSt
{...}, viewState: {...}, scopeState: {...}, urlState: {...}, setupState: {...}, renderedState: {...}, commands: {...}
null, children: undefined}>
<Canvas {canvas: {...}, width: 2306, height: 1460, dpi: 2, magnify: null, canvasRef: {...}, children: {...}
<AutoSize {canvas: {...}, children: children(...)/*#__PURE__*/ _jsxDEV(Canvas, { canvas: ..., key: null
<Inspectable {container: {...}, children: {...}, key: null}>
Inspect all state
Line 3
Y Axis
Line 5
Line 1


Walls
test
IDENTITY
Ground Level
Local undo/redo
+ Multiplayer ?
Line 7
Line 8
X Axis
Y Axis
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
MultiGather
View
PanControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
Props
Fiber
Provide(TransformContext)
context {initialValue: {...}, displayName: TransformContext, context: true}
value Object
size [1153, 730]
matrix Float32Array{0: 18.325485229492188, 1: 0, 2: 0, 3: -18.325485229492188, 4: 63
inverse Float32Array{0: 0.054569, 1: 0, 2: 0, 3: -0.054569, 4: -34.73160934448242, 5: 20
layout [-34.73160934448242, -19.222892828285694, 28.18623547628522, 20.612342
travel [-184, 606]
Yeeted
reduced {select: [un
draw: [{...}, {...}, {.....
Rendered By
<Transform {x: 35.887
zoom: 19.49008359527555, t
travelY: 460, children: {...
<PanControls {centered: true, x: 6.308065651556673, y: 7.75350789874147, zoom: 19.49008
version: 87, onChange: handleChange(...)setViewTarget2D(viewType, x, y, zoom), minZoom: 1, max
children: children(...)/*#__PURE__*/ _jsxDEV(Transform, { x: x,..., key: null}>
<View {width: 1153, height: 730, canvasRef: {...}, store: {...}, assetLibrary: {...}, materialLibrary: {...},
{...}, preferencesState: {...}, selectionState: {...}, unitState: {...}, treeState: {...}, baseState: {...}, stagingSt
{...}, viewState: {...}, scopeState: {...}, urlState: {...}, setupState: {...}, renderedState: {...}, commands: {...}
null, children: undefined}>
<Canvas {canvas: {...}, width: 2306, height: 1460, dpi: 2, magnify: null, canvasRef: {...}, children: {...}
<AutoSize {canvas: {...}, children: children(...)/*#__PURE__*/ _jsxDEV(Canvas, { canvas: ..., key: null
<Inspectable {container: {...}, children: {...}, key: null}>
Inspect all state
Line 3
Y Axis
Line 5
Line 1


Walls
test
IDENTITY
Ground Level
Local undo/redo
+ Multiplayer ?
= Any value can
change any time
Base
Wall 1
Wall 2
Wall 3
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
Yeet
Props
Fiber
Provide(TransformContext)
context {initialValue: {...}, displayName: TransformContext, context: true}
value Object
size [1153, 730]
matrix Float32Array{0: 18.325485229492188, 1: 0, 2: 0, 3: -18.325485229492188, 4: 63
inverse Float32Array{0: 0.054569, 1: 0, 2: 0, 3: -0.054569, 4: -34.73160934448242, 5: 20
layout [-34.73160934448242, -19.222892828285694, 28.18623547628522, 20.612342
travel [-184, 606]
Yeeted
reduced {select: [un
draw: [{...}, {...}, {.....
Rendered By
<Transform {x: 35.887
zoom: 19.49008359527555, t
travelY: 460, children: {...
<PanControls {centered: true, x: 6.308065651556673, y: 7.75350789874147, zoom: 19.49008
version: 87, onChange: handleChange(...)setViewTarget2D(viewType, x, y, zoom), minZoom: 1, max
children: children(...)/*#__PURE__*/ _jsxDEV(Transform, { x: x,..., key: null}>
<View {width: 1153, height: 730, canvasRef: {...}, store: {...}, assetLibrary: {...}, materialLibrary: {...},
{...}, preferencesState: {...}, selectionState: {...}, unitState: {...}, treeState: {...}, baseState: {...}, stagingSt
{...}, viewState: {...}, scopeState: {...}, urlState: {...}, setupState: {...}, renderedState: {...}, commands: {...}
null, children: undefined}>
<Canvas {canvas: {...}, width: 2306, height: 1460, dpi: 2, magnify: null, canvasRef: {...}, children: {...}
<AutoSize {canvas: {...}, children: children(...)/*#__PURE__*/ _jsxDEV(Canvas, { canvas: ..., key: null
<Inspectable {container: {...}, children: {...}, key: null}>
Inspect all
state
Line 3
Y Axis
Line 5
Line 1


Walls
test
IDENTITY
Ground Level
Local undo/redo
+ Multiplayer ?
= Any value can
change any time
This is why you
should know React
Inspect all
state
Props
Fiber
Provide(TransformContext)
context {initialValue: {...}, displayName: TransformContext, context: true}
value Object
size [1153, 730]
matrix Float32Array{0: 18.325485229492188, 1: 0, 2: 0, 3: -18.325485229492188, 4: 63
inverse Float32Array{0: 0.054569, 1: 0, 2: 0, 3: -0.054569, 4: -34.73160934448242, 5: 20
layout [-34.73160934448242, -19.222892828285694, 28.18623547628522, 20.612342
travel [-184, 606]
Yeeted
reduced {select: [un
draw: [{...}, {...}, {.....
Rendered By
<Transform {x: 35.887
zoom: 19.49008359527555, t
travelY: 460, children: {..
<PanControls {centered: true, x: 6.308065651556673, y: 7.75350789874147, zoom: 19.49008
version: 87, onChange: handleChange(...)setViewTarget2D(viewType, x, y, zoom), minZoom: 1, max
children: children(...)/*#__PURE__*/ _jsxDEV(Transform, { x: x,..., key: null}>
<View {width: 1153, height: 730, canvasRef: {...}, store: {...}, assetLibrary: {...}, materialLibrary: {...},
{...}, preferencesState: {...}, selectionState: {...}, unitState: {...}, treeState: {...}, baseState: {...}, stagingSt
{...}, viewState: {...}, scopeState: {...}, urlState: {...}, setupState: {...}, renderedState: {...}, commands: {...}
null, children: undefined}>
<Canvas {canvas: {...}, width: 2306, height: 1460, dpi: 2, magnify: null, canvasRef: {...}, children: {...}
<AutoSize {canvas: {...}, children: children(...)/*#__PURE__*/ _jsxDEV(Canvas, { canvas: ..., key: null
<Inspectable {container: {...}, children: {...}, key: null}>
Size
as
Provide(TransformContext)
iGather
iew
anControls
Yeet
Yeet
Yeet
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
X Axis
Line 6
Line 3
Y Axis
Line 5
Line 1

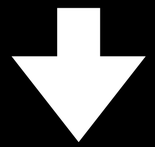

Walls
test
IDENTITY
Ground Level
Local undo/redo
+ Multiplayer ?
= Any value can
change any time
This is why you
should know React
Wall 4
Wall 5
Wall 6
Wall 7
Door 1
Door 2
Ground Level
Level 1
Level 2
Staging
Tour
NO ERRORS
Transform
Provide(TransformContext)
Provide(ProjectContext)
Memo(InnerView)
Fragment
Memo(SelectionUI)
ReferencePolygon
Memo(Polygon)
Inspector =
React in Live
in React
Props
Fiber
Provide(TransformContext)
context {initialValue: {...}, displayName: TransformContext, context: true}
value Object
size [1153, 730]
matrix Float32Array{0: 18.325485229492188, 1: 0, 2: 0, 3: -18.325485229492188, 4: 63
inverse Float32Array{0: 0.054569, 1: 0, 2: 0, 3: -0.054569, 4: -34.73160934448242, 5: 20
layout [-34.73160934448242, -19.222892828285694, 28.18623547628522, 20.612342
travel [-184, 606]
Yeeted
reduced {select: [un
draw: [{...}, {...}, {.....
Rendered By
<Transform {x: 35.887
zoom: 19.49008359527555, t
travelY: 460, children: {...
<PanControls {centered: true, x: 6.308065651556673, y: 7.75350789874147, zoom: 19.49008
version: 87, onChange: handleChange(...)setViewTarget2D(viewType, x, y, zoom), minZoom: 1, ma
children: children(...)/*#__PURE__*/ _jsxDEV(Transform, { x: x,..., key: null}>
<View {width: 1153, height: 730, canvasRef: {...}, store: {...}, assetLibrary: {...}, materialLibrary: {...},
{...}, preferencesState: {...}, selectionState: {...}, unitState: {...}, treeState: {...}, baseState: {...}, stagingSt
{...}, viewState: {...}, scopeState: {...}, urlState: {...}, setupState: {...}, renderedState: {...}, commands: {...}
null, children: undefined}>
<Canvas {canvas: {...}, width: 2306, height: 1460, dpi: 2, magnify: null, canvasRef: {...}, children: {...}
<AutoSize {canvas: {...}, children: children(...)/*#__PURE__*/ _jsxDEV(Canvas, { canvas: ..., key: null
<Inspectable {container: {...}, children: {...}, key: null}>
Inspect all
state
Line 3
Y Axis
Line 5
Line 1

```ts
export type TransformProps = {
  x?: number,
  y?: number,
  zoom?: number,
  travelX?: number,
  travelY?: number,
};

export const Transform: LC<TransformProps> = (props: PropsWithChildren<TransformProps>) => {
  const {x, y, zoom, travelX, travelY, children} = props;
  const {size, layout: l, matrix: m, inverse: i} = useContext(TransformContext);

  const [matrix, inverse] = useMemo(() => [
    new Float32Array([
      zoom, 0,
      0, -zoom,
      x * zoom, y * zoom,
    ]),
    new Float32Array([
      1/zoom, 0,
      0, -1/zoom,
      -x, y,
    ]),
  ], [x, y, zoom]);

  const transformContext = useMemo(() => {
    const cm = new Float32Array(6);
```

```ts
export type TransformProps = {
  x?: number,
  y?: number,
  zoom?: number,
  travelX?: number,
  travelY?: number,
};

export const Transform: LC<TransformProps> = (props: PropsWithChildren<TransformProps>) => {
  const {x, y, zoom, travelX, travelY, children} = props;
  const {size, layout: l, matrix: m, inverse: i} = useContext(TransformContext);

  const [matrix, inverse] = useMemo(() => [
    new Float32Array([
      zoom, 0,
      0, -zoom,
      x * zoom, y * zoom,
    ]),
    new Float32Array([
      1/zoom, 0,
      0, -1/zoom,
      -x, y,
    ]),
  ], [x, y, zoom]);

  const transformContext = useMemo(() => {
    const cm = new Float32Array(6);
```

Looks just like React

```ts
export type TransformProps = {
  x?: number,
  y?: number,
  zoom?: number,
  travelX?: number,
  travelY?: number,
};
```



Looks just like React

```ts
export const Transform: LC<TransformProps> = (props: PropsWithChildren<TransformProps>) => {
  const {x, y, zoom, travelX, travelY, children} = props;
  const {size, layout: l, matrix: m, inverse: i} = useContext(TransformContext);

  const [matrix, inverse] = useMemo(() => [
    new Float32Array([
      zoom, 0,
      0, -zoom,
      x * zoom, y * zoom,
    ]),
    new Float32Array([
      1/zoom, 0,
      0, -1/zoom,
      -x, y,
    ]),
  ], [x, y, zoom]);

  const transformContext = useMemo(() => {
    const cm = new Float32Array(6);
```

```ts
export type TransformProps = {
  x?: number,
  y?: number,
  zoom?: number,
  travelX?: number,
  travelY?: number,
};
```

Looks just like React

```ts
export const Transform: LC<TransformProps> = (props: PropsWithChildren<TransformProps>) => {
  const {x, y, zoom, travelX, travelY, children} = props;
  const {size, layout: l, matrix: m, inverse: i} = useContext(TransformContext);

  const [matrix, inverse] = useMemo(() => [
    new Float32Array([
      zoom, 0,
      0, -zoom,
      x * zoom, y * zoom,
    ]),
    new Float32Array([
      1/zoom, 0,
      0, -1/zoom,
      -x, y,
    ]),
  ], [x, y, zoom]);

  const transformContext = useMemo(() => {
    const cm = new Float32Array(6);
```

```ts
out.push(
  <Polyline
    key={key}
    opacity={opacity}
    path={path as XY[]}
    paths={paths as XY[][]}
    {...style.line}
    selectable={selectable}
    zIndex={Z_INDICES.line + style.z}
  />
);
```

```ts
export type TransformProps = {
  x?: number,
  y?: number,
  zoom?: number,
  travelX?: number,
  travelY?: number,
};
```

Looks just like React

```ts
export const Transform: LC<TransformProps> = (props: PropsWithChildren<TransformProps>) => {
  const {x, y, zoom, travelX, travelY, children} = props;
  const {size, layout: l, matrix: m, inverse: i} = useContext(TransformContext);

  const [matrix, inverse] = useMemo(() => [
    new Float32Array([
      zoom, 0,
      0, -zoom,
      x * zoom, y * zoom,
    ]),
    new Float32Array([
      1/zoom, 0,
      0, -1/zoom,
      -x, y,
    ]),
  ], [x, y, zoom]);

  const transformContext = useMemo(() => {
    const cm = new Float32Array(6);
```

So just do the same
for WebGPU?

```ts
out.push(
  <Polyline
    key={key}
    opacity={opacity}
    path={path as XY[]}
    paths={paths as XY[][]}
    {...style.line}
    selectable={selectable}
    zIndex={Z_INDICES.line + style.z}
  />
);
```

Graphics vs Web

Graphics

Web

Graphics

Driven by

Web

Graphics

Driven by

- AAA video games

Web

Graphics

Driven by

- AAA video games
- Industrial CAD

Web

Graphics

Driven by

- AAA video games
- Industrial CAD

- Large teams

Web

Graphics

Driven by

- AAA video games
- Industrial CAD

- Large teams
- Offline delivery

Web

Graphics

Driven by

- AAA video games
- Industrial CAD

- Large teams
- Offline delivery
- Rendering performance

Web

Graphics

Driven by

- AAA video games
- Industrial CAD

- Large teams
- Offline delivery
- Rendering performance

Monolithic projects

Web

Graphics

Driven by

- AAA video games
- Industrial CAD

- Large teams
- Offline delivery
- Rendering performance

Monolithic projects

Web

Driven by

Graphics

Driven by

- AAA video games
- Industrial CAD

- Large teams
- Offline delivery
- Rendering performance

Monolithic projects

Web

Driven by

- SaaS companies

Graphics

Driven by

- AAA video games
- Industrial CAD

- Large teams
- Offline delivery
- Rendering performance

Monolithic projects

Web

Driven by

- SaaS companies
- Browsers & Open Source

Graphics

Driven by

- AAA video games
- Industrial CAD

- Large teams
- Offline delivery
- Rendering performance

Monolithic projects

Web

Driven by

- SaaS companies
- Browsers & Open Source

- Small teams

Graphics

Driven by

- AAA video games
- Industrial CAD

- Large teams
- Offline delivery
- Rendering performance

Monolithic projects

Web

Driven by

- SaaS companies
- Browsers & Open Source

- Small teams
- Continuous delivery

Graphics

Driven by

- AAA video games
- Industrial CAD

- Large teams
- Offline delivery
- Rendering performance

Monolithic projects

Web

Driven by

- SaaS companies
- Browsers & Open Source

- Small teams
- Continuous delivery
- Compatibility

Graphics

Driven by

- AAA video games
- Industrial CAD

- Large teams
- Offline delivery
- Rendering performance

Monolithic projects

Web

Driven by

- SaaS companies
- Browsers & Open Source

- Small teams
- Continuous delivery
- Compatibility

Composition and reuse

Graphics

Driven by

- AAA video games
- Industrial CAD

- Large teams
- Offline delivery
- Rendering performance

Web

Driven by

- SaaS companies
- Browsers & Open Source

- Small teams
- Continuous delivery
- Compatibility

Monolithic projects

Two completely different worlds

Composition and reuse

"How do I draw a f😿ing line"

Graphics **Web**

"How do I draw a f😿ing line"

Use **canvas**

Graphics

Web

"How do I draw a f😿ing line"

Use **canvas**

```
context.lineWidth = 5
context.moveTo(…)
context.lineTo(…)
context.stroke()
```

Graphics

Web

"How do I draw a f😿ing line"

Graphics **Web**

"How do I draw a f😿ing line"

Make a
`render target`

Graphics **Web**

"How do I draw a f😿ing line"

Make a
`render target`

What's a
`render target`?

Graphics

Web

"How do I draw a f😿ing line"

Make a
`render target`

Fill a
`vertexBuffer`
with **`triangles`**

What's a
`render target`?

Graphics **Web**

"How do I draw a f😿ing line"

Make a
`render target`

Fill a
`vertexBuffer`
with **`triangles`**

What's a
`render target`?

But I want
to draw **lines**

Graphics

Web

"How do I draw a f😿ing line"

Make a
`render target`

Learn a
`shader` language

Fill a
`vertexBuffer`
with **`triangles`**

What's a
`render target`?

But I want
to draw **lines**

Graphics **Web**

"How do I draw a f😿ing line"

Make a
`render target`

Learn a
`shader` language

Fill a
`vertexBuffer`
with **`triangles`**

What's a
`render target`?

But I want
to draw **lines**

Which one?

Graphics

Web

"How do I draw a f😿ing line"

Make a
`render target`

Learn a
`shader` language

Fill a
`vertexBuffer`
with **`triangles`**

Compile a
`vertex`/`fragment`
shader...

What's a
`render target`?

Which one?

But I want
to draw **lines**

Graphics

Web

"How do I draw a f😿ing line"

Make a **`render target`**

Fill a **`vertexBuffer`** with **`triangles`**

Learn a **`shader`** language

Compile a **`vertex`**/**`fragment`** shader...

Build a **`pipeline`** with **`descriptors`**

What's a **`render target`**?

But I want to draw **lines**

Which one?

Graphics

Web

"How do I draw a f😿ing line"

Make a
`render target`

Learn a
`shader` language

Fill a
`vertexBuffer`
with **`triangles`**

Build a
`pipeline` with
`descriptors`

Compile a
`vertex/fragment`
shader...

...and **make a**
`render` **loop**
to dispatch it

What's a
`render target`?

But I want
to draw **lines**

Which one?

Graphics

Web

"How do I draw a f😿ing line"

Make a
`render target`

Learn a
`shader` language

Build a
`pipeline` with
`descriptors`

Fill a
`vertexBuffer`
with **`triangles`**

Compile a
`vertex/fragment`
shader...

...and **make a**
`render` **loop**
to dispatch it

Graphics

What's a
`render target`?

But I want
to draw **lines**

Which one?

...

Web

"How do I draw a f😿ing line"

Graphics **Web**

"How do I draw a f😿ing line"

I get a
black screen

Graphics

Web

"How do I draw a f😿ing line"

I get a
black screen

Where is the
debugger?

Graphics

Web

"How do I draw a f😿ing line"

There is no **debugger**

I get a
black screen

Where is the
debugger?

Graphics

Web

"How do I draw a f😿ing line"

There is no **`debugger`**

And no **`printf()`**

Graphics

I get a
black screen

Where is the
`debugger`?

Web

Graphics

Applications

Web

Applications

Graphics

Applications

- Multiplayer

Web

Applications

- Multiplayer

Graphics

Applications

- Multiplayer
- 2D and 3D content editors

Web

Applications

- Multiplayer
- 2D and 3D content editors

Graphics

Applications

- Multiplayer
- 2D and 3D content editors

- 2-3 year project turnover

Web

Applications

- Multiplayer
- 2D and 3D content editors

- 2-3 year project turnover

Graphics

Applications

- Multiplayer
- 2D and 3D content editors

- 2-3 year project turnover
- Continuous bugfixes

Web

Applications

- Multiplayer
- 2D and 3D content editors

- 2-3 year project turnover
- Continuous bugfixes

Graphics

Applications

- Multiplayer
- 2D and 3D content editors

- 2-3 year project turnover
- Continuous bugfixes
- Decades of legacy cruft

Web

Applications

- Multiplayer
- 2D and 3D content editors

- 2-3 year project turnover
- Continuous bugfixes
- Decades of legacy cruft

Graphics

Applications

- Multiplayer
- 2D and 3D content editors

- 2-3 year project turnover
- Continuous bugfixes
- Decades of legacy cruft
- Only bearable with extra tooling

Web

Applications

- Multiplayer
- 2D and 3D content editors

- 2-3 year project turnover
- Continuous bugfixes
- Decades of legacy cruft
- Only bearable with extra tooling

Graphics

Applications

- Multiplayer
- 2D and 3D content editors

- 2-3 year project turnover
- Continuous bugfixes
- Decades of legacy cruft
- Only bearable with extra tooling

Web

Install **node** and **npm**

Applications

- Multiplayer
- 2D and 3D content editors

- 2-3 year project turnover
- Continuous bugfixes
- Decades of legacy cruft
- Only bearable with extra tooling

Graphics

Applications

- Multiplayer
- 2D and 3D content editors

- 2-3 year project turnover
- Continuous bugfixes
- Decades of legacy cruft
- Only bearable with extra tooling

Web

Install **`node`** and **`npm`**

Make a **`package.json`**

Applications

- Multiplayer
- 2D and 3D content editors

- 2-3 year project turnover
- Continuous bugfixes
- Decades of legacy cruft
- Only bearable with extra tooling

Graphics

Web

Graphics

Web

WebGPU
is kinda
shit

Graphics

WebGPU
is kinda
shit

Web

Yeah,
browsers
are kinda
shit

Graphics

WebGPU
is kinda
shit

Vulkan
is worse

Web

Yeah,
browsers
are kinda
shit

Graphics

WebGPU
is kinda
shit

Vulkan
is worse

Web

Yeah,
browsers
are kinda
shit

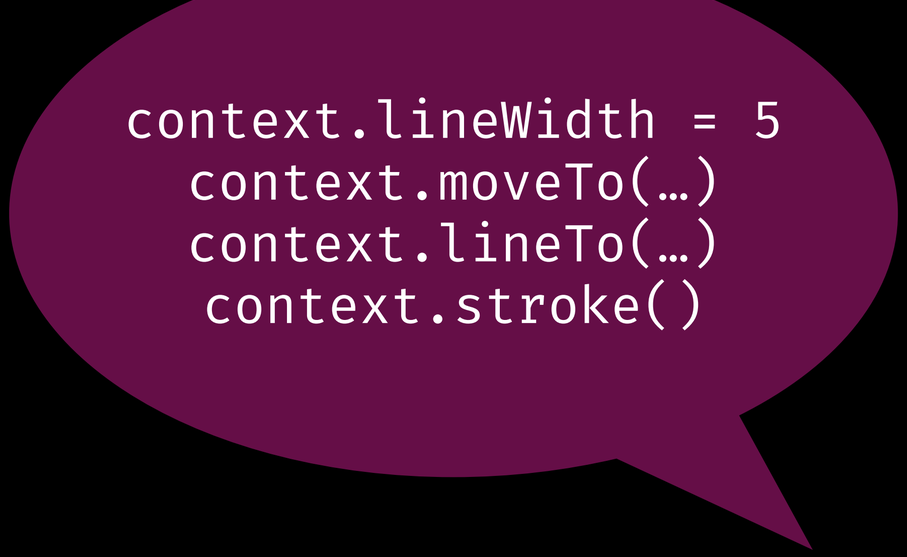
So is our
back-end

What went wrong
with GPUs

Immediate Mode

Immediate Mode

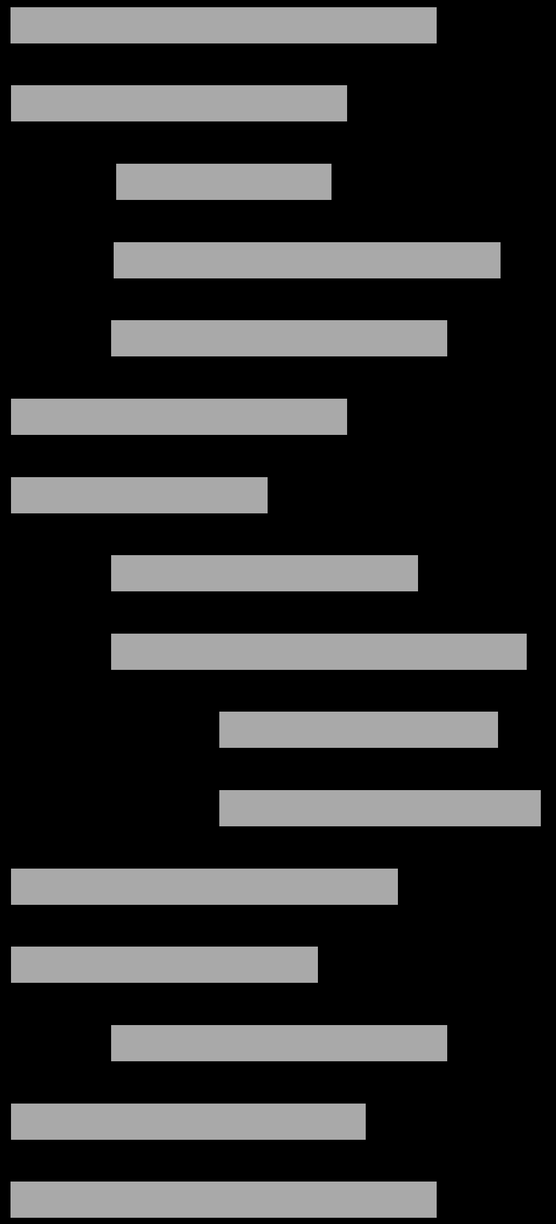
- Write all the code to produce 1 frame



Immediate Mode

- Write all the code to produce 1 frame

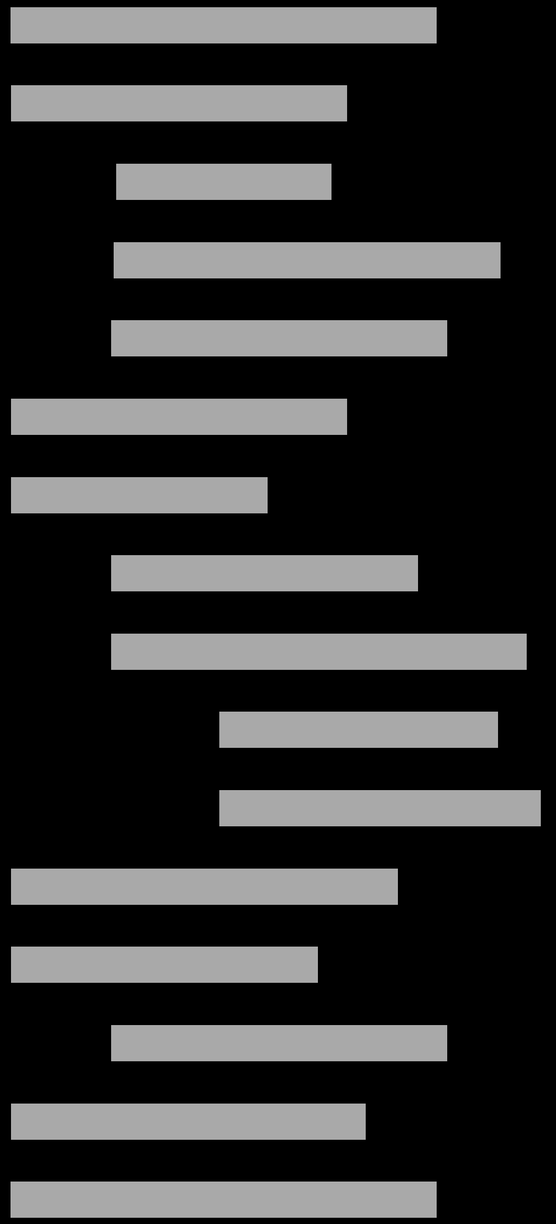
```
context.lineWidth = 5
  context.moveTo(…)
  context.lineTo(…)
   context.stroke()
```



Immediate Mode

- Write all the code to produce 1 frame
- Call it again with new input

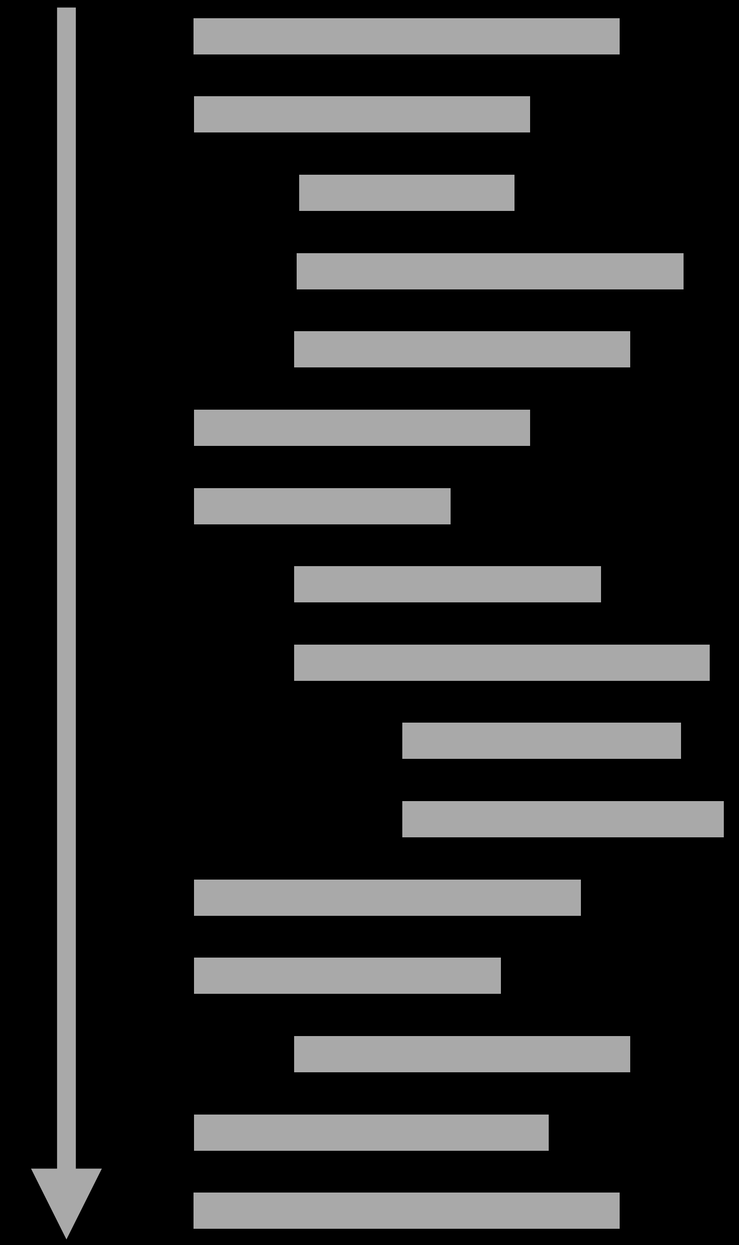
```
context.lineWidth = 5
  context.moveTo(…)
  context.lineTo(…)
   context.stroke()
```



Immediate Mode

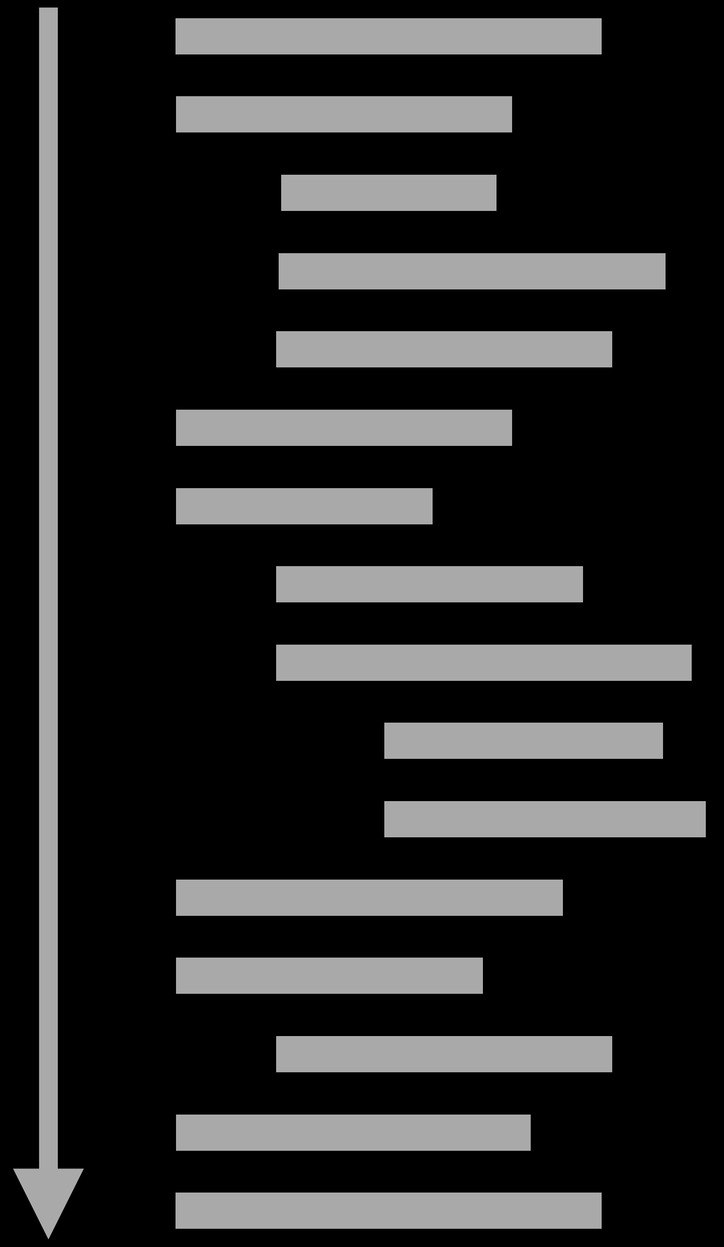
- Write all the code to produce 1 frame
- Call it again with new input

```
context.lineWidth = 5
  context.moveTo(…)
  context.lineTo(…)
   context.stroke()
```



Immediate Mode

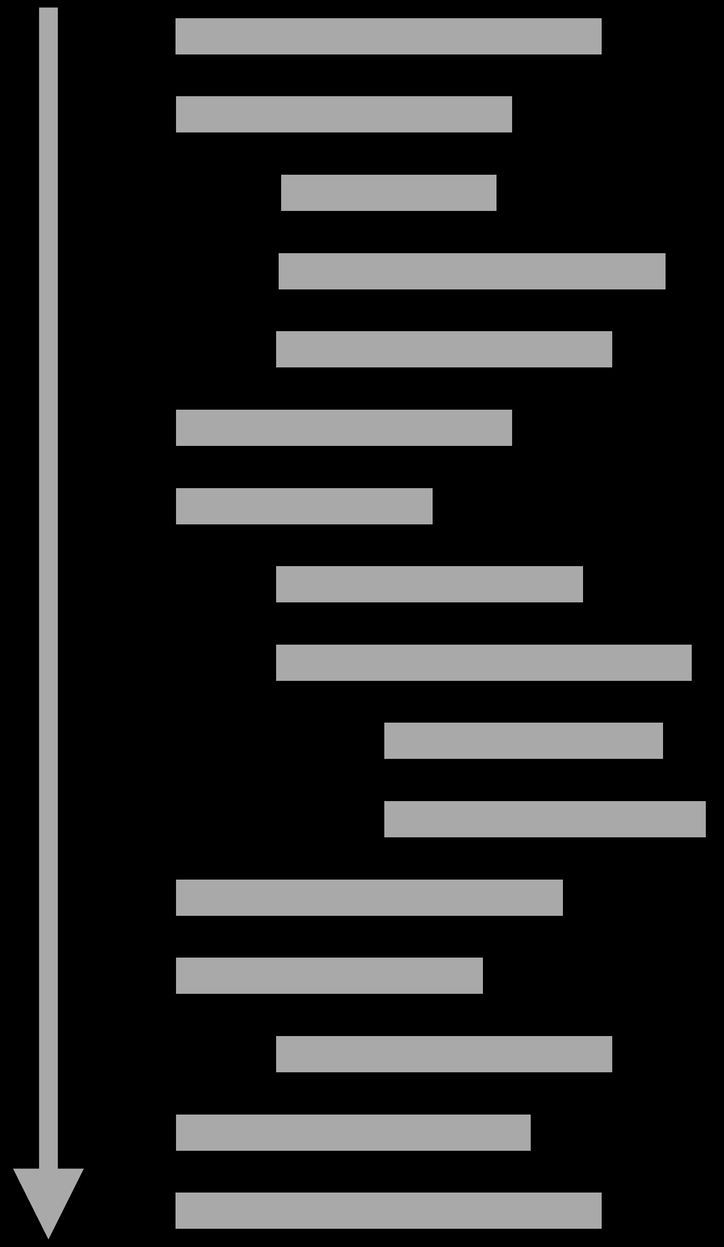
- Write all the code to produce 1 frame
- Call it again with new input



Immediate Mode

- Write all the code to produce 1 frame
- Call it again with new input

✓ Very simple

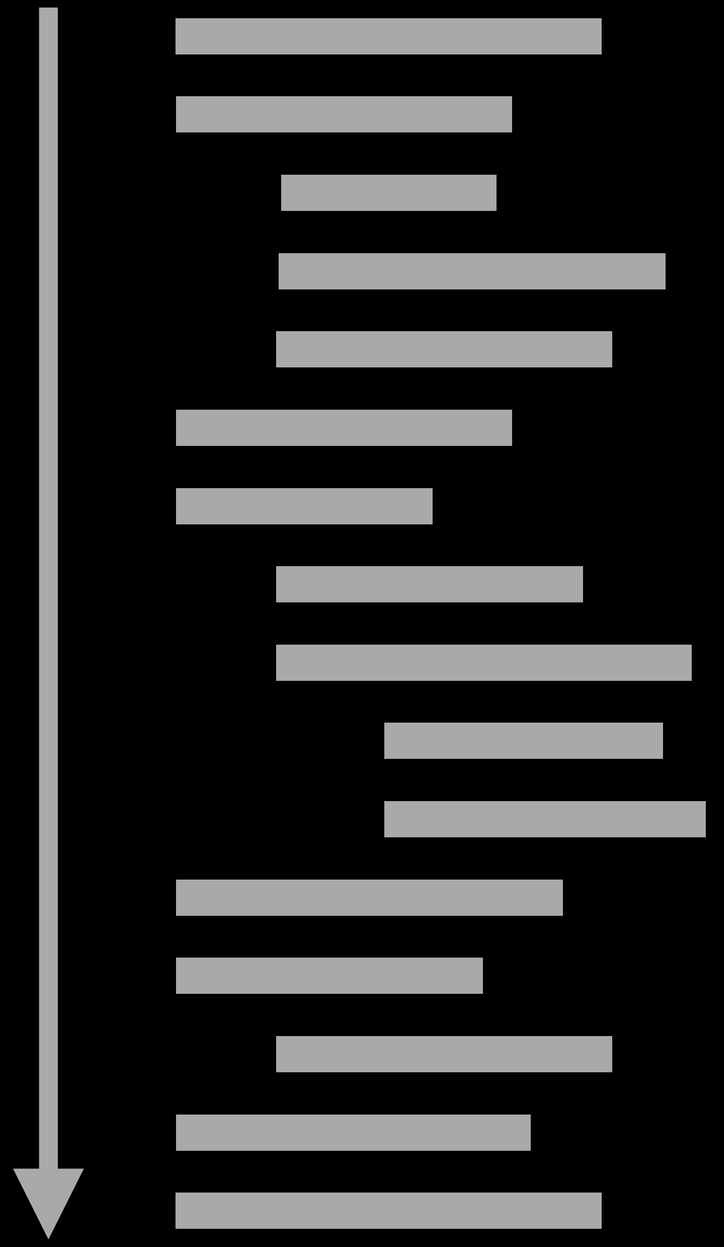


Immediate Mode

- Write all the code to produce 1 frame
- Call it again with new input

✓ Very simple

✗ Doesn't scale

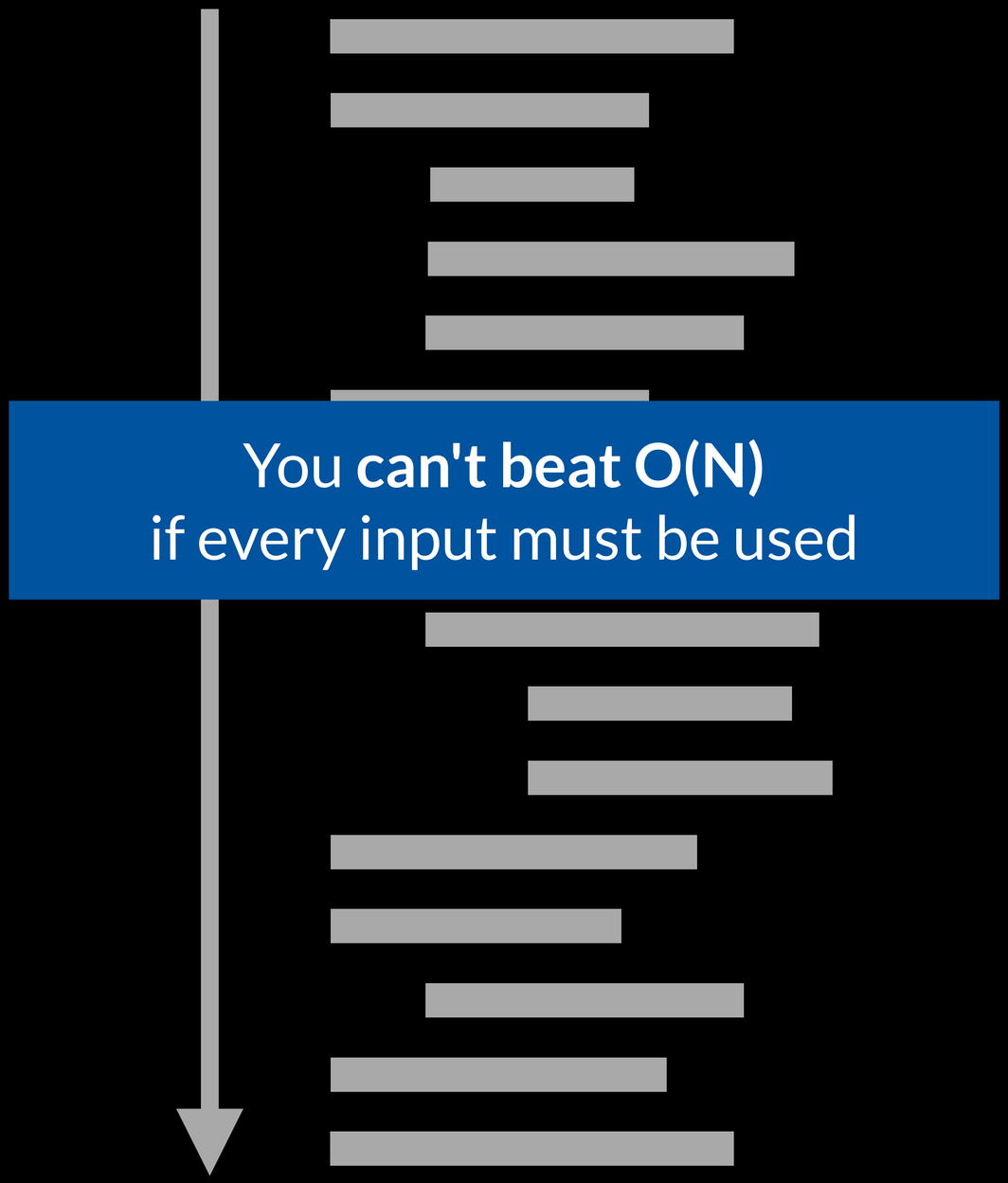


Immediate Mode

- Write all the code to produce 1 frame
- Call it again with new input

✓ Very simple

✗ Doesn't scale

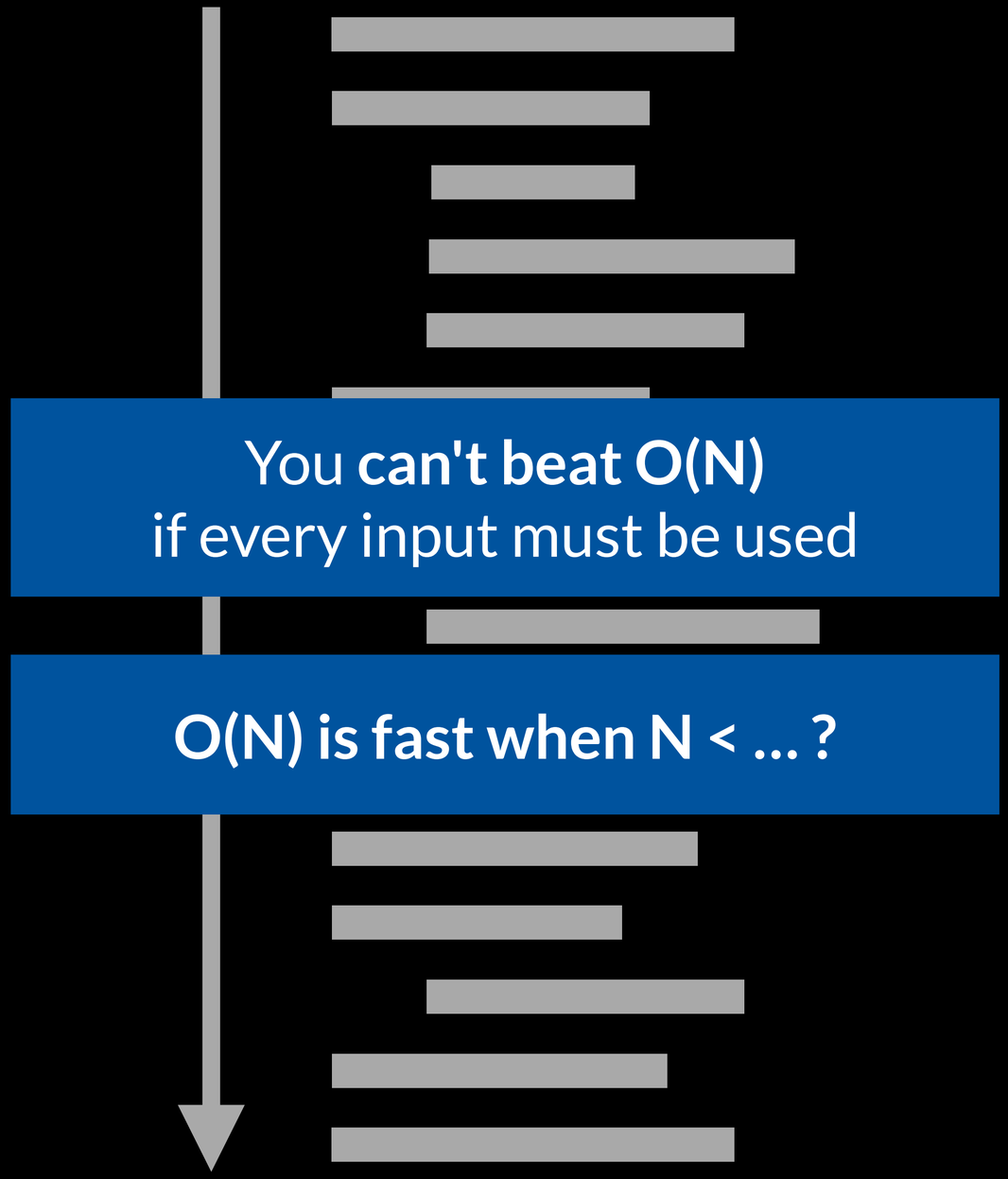


Immediate Mode

- Write all the code to produce 1 frame
- Call it again with new input

✓ Very simple

✗ Doesn't scale



Retained Mode

On **GPU**?

Retained Mode

On **GPU**?

```
fn ( a, b, c, d )
```

Retained Mode

On **GPU**?

- Allocate GPU memory

```
fn ( a, b, c, d )
```

Retained Mode

On **GPU**?

- Allocate GPU memory
- Upload data

```
fn ( a, b, c, d )
```

Retained Mode

On **GPU**?

- Allocate GPU memory
- Upload data
- Compile shader

```
fn ( a, b, c, d )
```

Retained Mode

On **GPU**?

- Allocate GPU memory
- Upload data
- Compile shader
- Dispatch

```
fn ( a, b, c, d )
```

Retained Mode

On **GPU**?

- Allocate GPU memory
- Upload data
- Compile shader
- Dispatch
- Await

```
fn ( a, b, c, d )
```

Retained Mode

On **GPU**?

- Allocate GPU memory
- Upload data
- Compile shader
- Dispatch
- Await

```
fn ( a, b, c, d )
```

"How do I call a f😿ing function"

Shader Dispatch

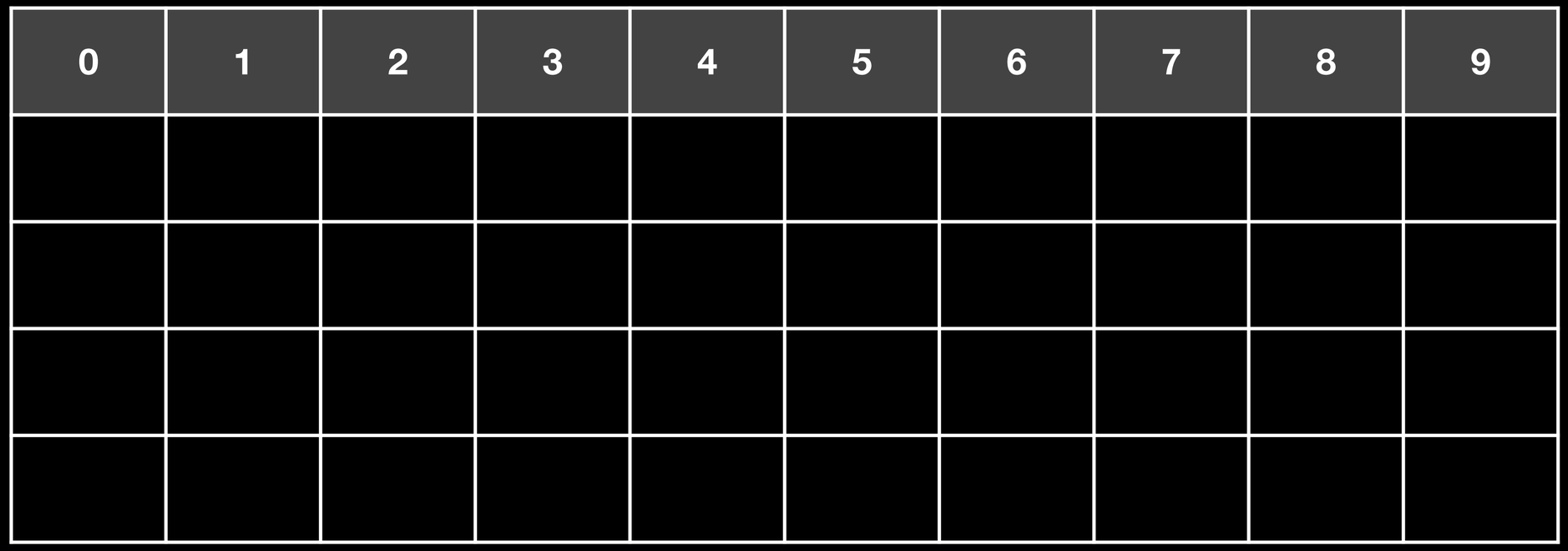
```
fn
(
  a
  b
  c
  d
)
```

Shader Dispatch

fn

a
b
c
d

Threads (N > 1,000,000)



Shader Dispatch

fn

a

b

c

d

Threads (N > 1,000,000)

0	1	2	3	4	5	6	7	8	9	…
3	4	1	3	2	5	7	8	2	3	
XYZ	**XYZ**	**XYZ**	**XYZ**	**XYZ**	**XYZ**	**XYZ**	**XYZ**	**XYZ**	**XYZ**	

Shader Dispatch

fn(*a*, *b*, *c*, *d*)

Threads (N > 1,000,000)

0	1	2	3	4	5	6	7	8	9	…
3	4	1	3	2	5	7	8	2	3	
XYZ	**XYZ**	**XYZ**	**XYZ**	**XYZ**	**XYZ**	**XYZ**	**XYZ**	**XYZ**	**XYZ**	

Memory bandwidth is the **main bottleneck**

Spread Policies

fn

a

b

c

d

Threads (N > 1,000,000)

0	1	2	3	4	5	6	7	8	9	…
3	4	1	3	2	5	7	8	2	3	

XYZ

Spread Policies

fn

1 to 1

b

c

d

Threads (N > 1,000,000)

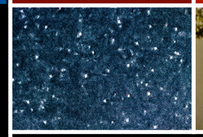
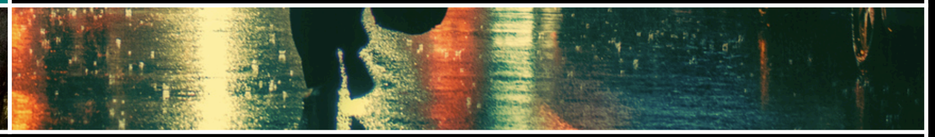
0	1	2	3	4	5	6	7	8	9	…
3	4	1	3	2	5	7	8	2	3	

XYZ

Spread Policies

fn

Threads (N > 1,000,000)

	0	1	2	3	4	5	6	7	8	9	…
1 to 1	3	4	1	3	2	5	7	8	2	3	
Repeat	0	1	2	3	4	0	1	2	3	4	
											
	XYZ										

c

d

Spread Policies

fn

Threads (N > 1,000,000)

	0	1	2	3	4	5	6	7	8	9	…
1 to 1	3	4	1	3	2	5	7	8	2	3	
Repeat	0	1	2	3	4	0	1	2	3	4	
Lookup	0	1	1	1	1	2	2	2	2	2	
	XYZ										

d

Spread Policies

fn

Threads (N > 1,000,000)

	0	1	2	3	4	5	6	7	8	9	…
1 to 1	3	4	1	3	2	5	7	8	2	3	
Repeat	0	1	2	3	4	0	1	2	3	4	
Lookup	0	1	1	1	1	2	2	2	2	2	
Constant	XYZ										

Spread Policies

fn

Threads (N > 1,000,000)

	0	1	2	3	4	5	6	7	8	9	…
1 to 1	Vertex					`array[i]`					
Repeat	Instanced					`array[i % n]`				`[i / n]`	
Lookup	Indexed					`array[lookup[i]]`					
Constant	Uniform					`array[0]`					

Spread Policies

+ Storage	`bigArray[randomAccess]`

fn

Threads (N > 1,000,000)

	0	1	2	3	4	5	6	7	8	9	…
1 to 1				Vertex		`array[i]`					
Repeat				Instanced		`array[i % n]`			`[i / n]`		
Lookup				Indexed		`array[lookup[i]]`					
Constant				Uniform		`array[0]`					

Spread Policies

+ Storage	`bigArray[randomAccess]`

fn

Threads (N > 1,000,000)

	0	1	2	3	4	5	6	
1 to 1				Vertex		`array[i]`		
Repeat				Instanced		`array[i % n]`		`[i / n]`
Lookup				Indexed		`array[lookup[i]]`		
Constant				Uniform		`array[0]`		

⚠ **Legacy Cruft**

Spread Policies

+ Storage `bigArray[randomAccess]`

fn

Threads (N > 1,000,000)

	0	1	2	3	4	5	6
1 to 1				Vertex		`array[i]`	
Repeat				Instanced		`array[i % n]`	`[i / n]`
Lookup				Indexed		`array[lookup[i]]`	
Constant				Uniform		`array[0]`	

⚠ **Legacy Cruft**

Completely different **types and APIs** for each on both CPU and GPU 🤬

Spread Policies

+ Storage `bigArray[randomAccess]`

+ Restrictions on control flow and grouping 🫠

fn

Threads (N > 1,000,000)

	0	1	2	3	4	5	6
1 to 1				Vertex		`array[i]`	
Repeat				Instanced		`array[i % n]`	`[i / n]`
Lookup				Indexed		`array[lookup[i]]`	
Constant				Uniform		`array[0]`	

⚠ **Legacy Cruft**

Completely different **types and APIs** for each on both CPU and GPU 🤬

Spread Policies

Problem #1

\+ Storage `bigArray[randomAccess]`

\+ Restrictions on control flow and grouping 🫠

fn

Threads (N > 1,000,000)

	0	1	2	3	4	5	6
1 to 1				Vertex		`array[i]`	
Repeat				Instanced		`array[i % n]`	`[i / n]`
Lookup				Indexed		`array[lookup[i]]`	
Constant				Uniform		`array[0]`	

⚠ **Legacy Cruft**

<u>Completely</u> different **types and APIs** for each on both CPU and GPU 🤬

Spread Policies

Problem #1

+ Storage `bigArray[randomAccess]`
+ Restrictions on control flow and grouping 🫠

fn

Threads (N > 1,000,000)

	0	1	2	3	4	5	6	7	8	9	…
1 to 1				Vertex		`array[i]`					
Repeat				Instanced		`array[i % n]`		`[i / n]`			
Lookup				Indexed		`array[lookup[i]]`					
Constant				Uniform		`array[0]`					

Completely different **types and APIs** for each on both CPU and GPU 🤬

Spread Policies

Problem #1

+ Storage `bigArray[randomAccess]`
+ Restrictions on control flow and grouping 🤐

fn

Threads (N > 1,000,000)

	5	6	7	8	9	…
1 to 1		`array[i]`				
Repeat		`array[i % n]`				
Lookup	Indexed	`array[lookup[i]]`				
Constant	Uniform	`array[0]`				

Batch <u>everything</u>

<u>Completely</u> different **types and APIs** for each on both CPU and GPU 🤬

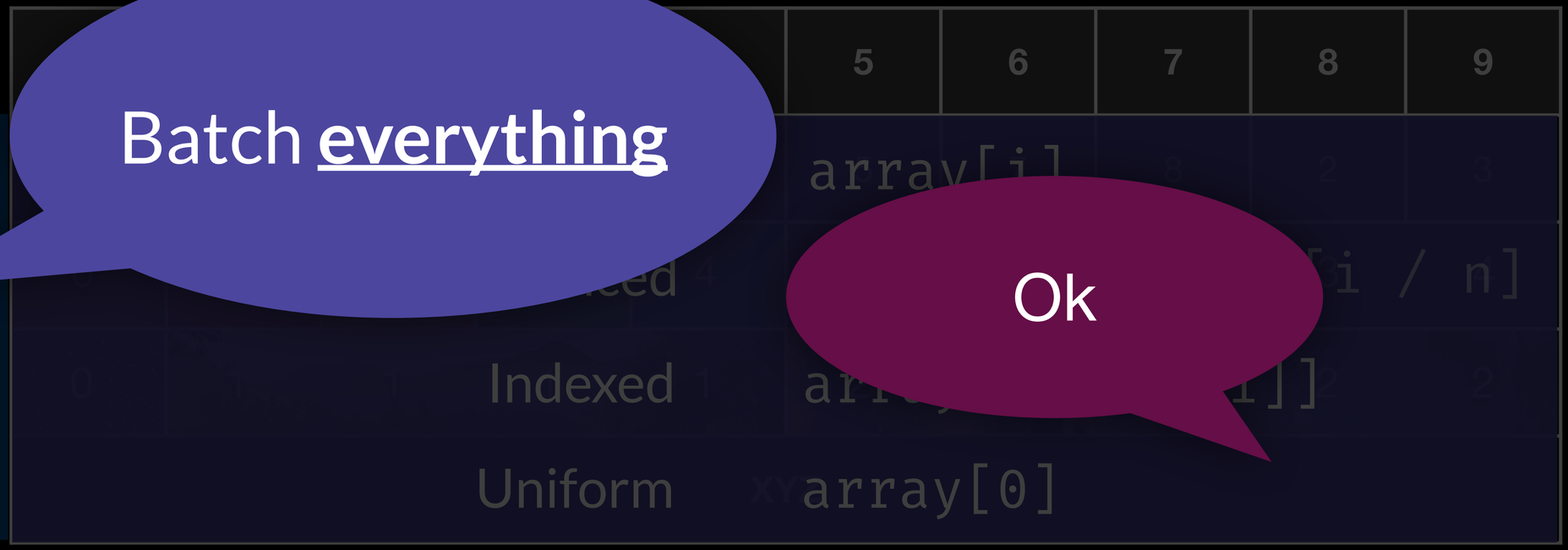
Spread Policies

Problem #1

+ Storage `bigArray[randomAccess]`
+ Restrictions on control flow and grouping 🤐

fn

Threads (N > 1,000,000)



Batch **everything**

Ok

Completely different **types and APIs** for each on both CPU and GPU 🤬

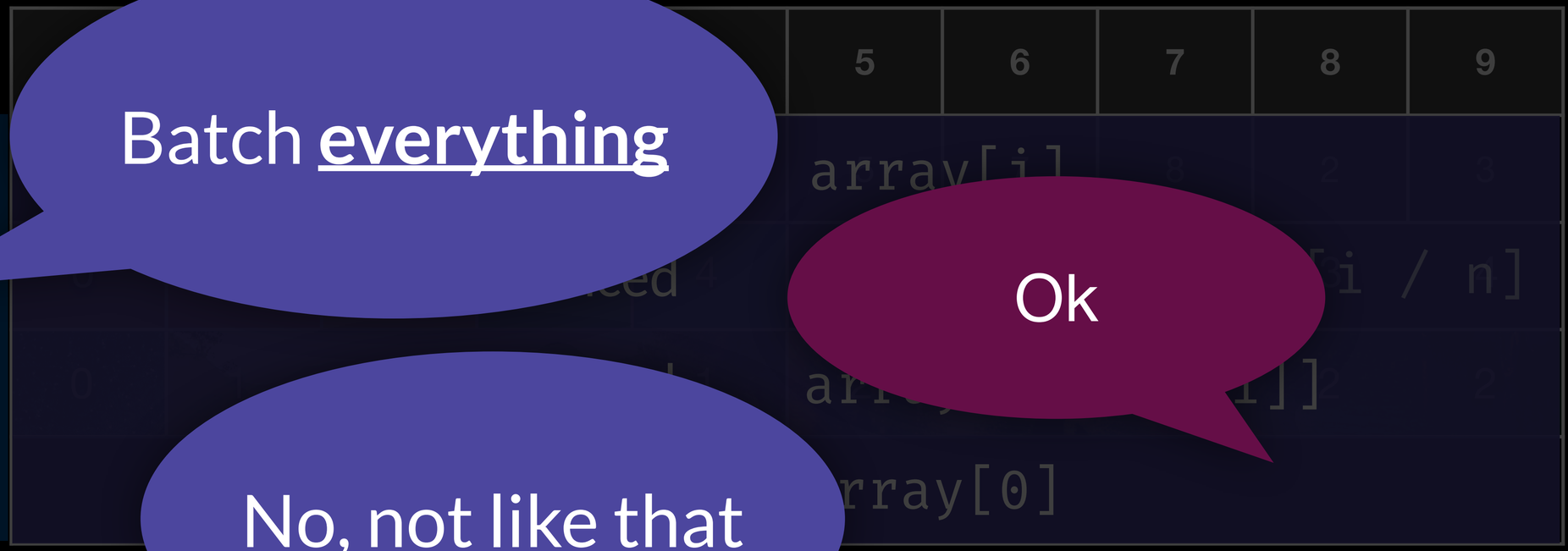
Spread Policies

Problem #1

+ Storage `bigArray[randomAccess]`
+ Restrictions on control flow and grouping 🤐

fn

Threads (N > 1,000,000)



1 to 1

Repeat

Lookup

Constant

Batch **everything**

Ok

No, not like that

Completely diff **is** for each on both CPU and GPU 🤬

Pipeline Dispatch

Pipeline Dispatch

```
fn ( input: Texture, output: Texture )
```

Pipeline Dispatch

```
fn ( input: Texture, output: Texture )
```

Map **every pixel** 1-to-1

Pipeline Dispatch

```
fn ( input: Texture, output: Texture )
```

Map **every pixel** 1-to-1

No spread

Pipeline Dispatch

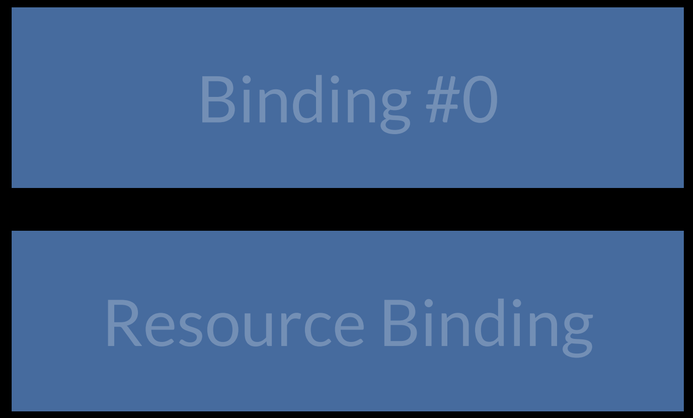
```
fn ( input: Texture, output: Texture )
```

Pipeline Dispatch

Resource Binding

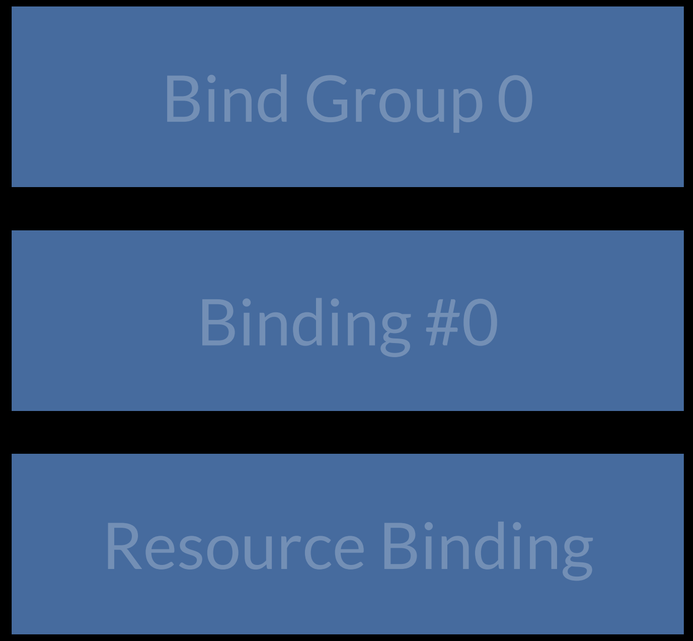
```
fn ( input: Texture, output: Texture )
```

Pipeline Dispatch




```
fn ( input: Texture, output: Texture )
```

Pipeline Dispatch

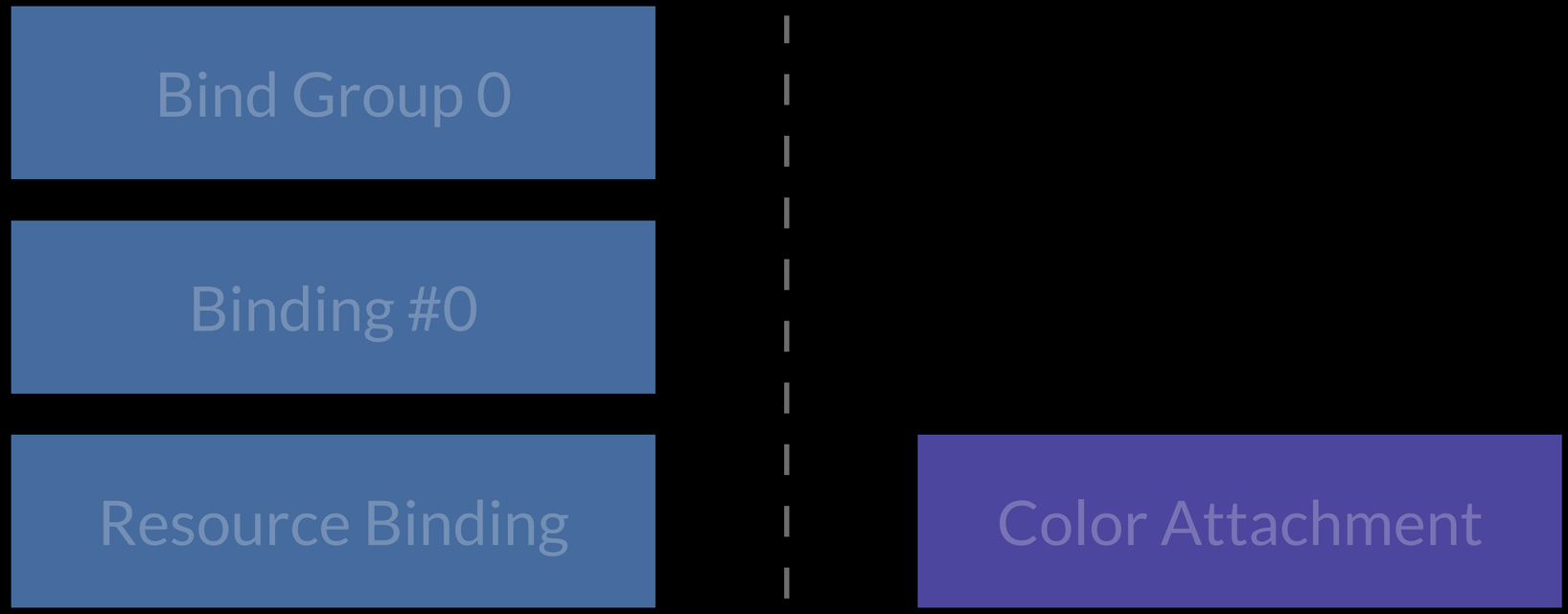



```
fn ( input: Texture, output: Texture )
```

Pipeline Dispatch

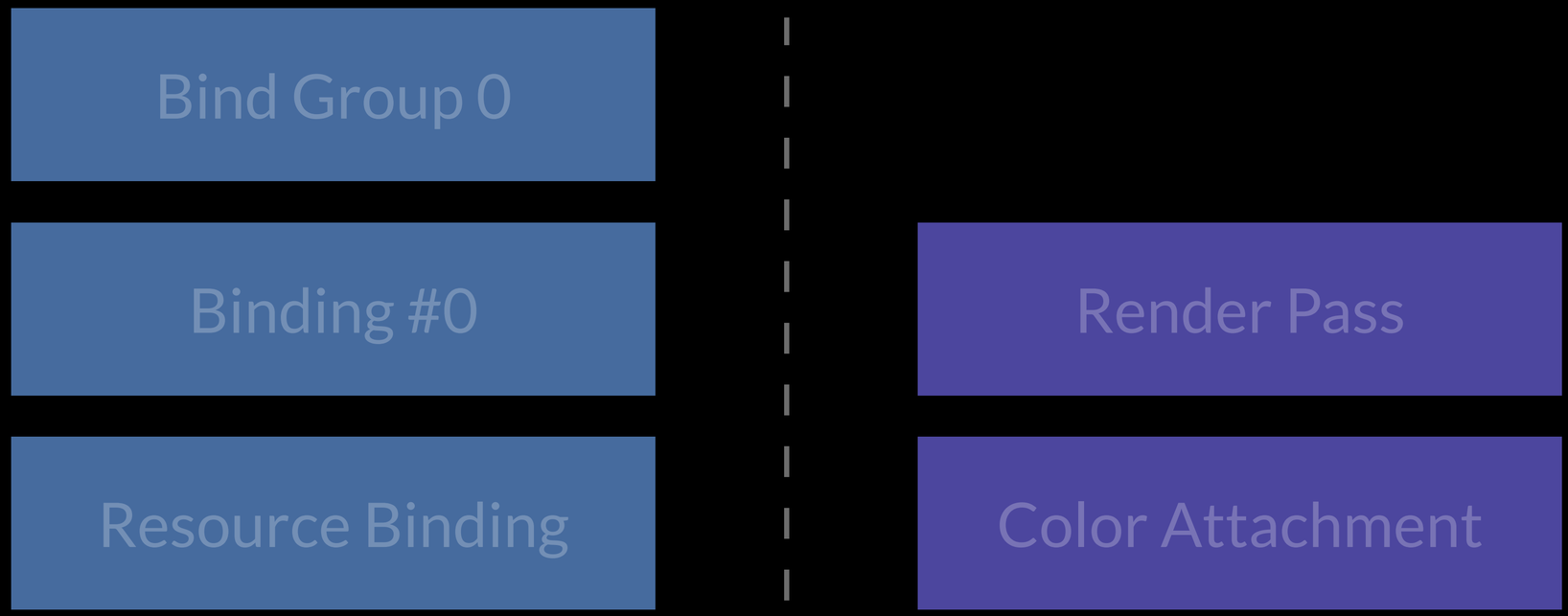

```
fn ( input: Texture, output: Target )
```

Pipeline Dispatch




```
fn ( input: Texture, output: Target )
```

Pipeline Dispatch




```
fn ( input: Texture, output: Target )
```

Binding #0

Resource Binding

Render Pass

Color Attachment

```
fn ( input: Texture, output: Target )
```

Binding #0

Render Pass

Resource Binding

Color Attachment

```
fn ( input: Texture, output: Target )
```

Color State

Binding #0

Render Pass

Resource Binding

Color Attachment

```
fn ( input: Texture, output: Target )
```

Color State

Pipeline

Binding #0

Render Pass

Resource Binding

Color Attachment

fn (input: Texture, output: Target)

Vertex Shader

Color State

Pipeline

Binding #0

Render Pass

Resource Binding

Color Attachment

fn (input: Texture, output: Target)

Vertex Shader

Color State

Fragment Shader

Pipeline

Binding #0

Resource Binding

Render Pass

Color Attachme

```
fn ( input: Texture, vertices: T[ ], output: Target )
```

Vertex Shader

Fragment Shader

Color State

Pipeline

Binding #0

Resource Binding

Render Pass

Color Attachme

```
fn ( input: Texture, vertices: T[ ], output: Target )
```

Vertex Shader

Fragment Shader

Vertex Buffer

Color State

Pipeline

Binding #0

Binding #1

Render Pass

Resource Binding

Sampler Binding

Color Attachme

```
fn ( input: Texture, vertices: T[ ], output: Target )
```

Vertex Shader

Vertex Buffer

Color State

Fragment Shader

Pipeline



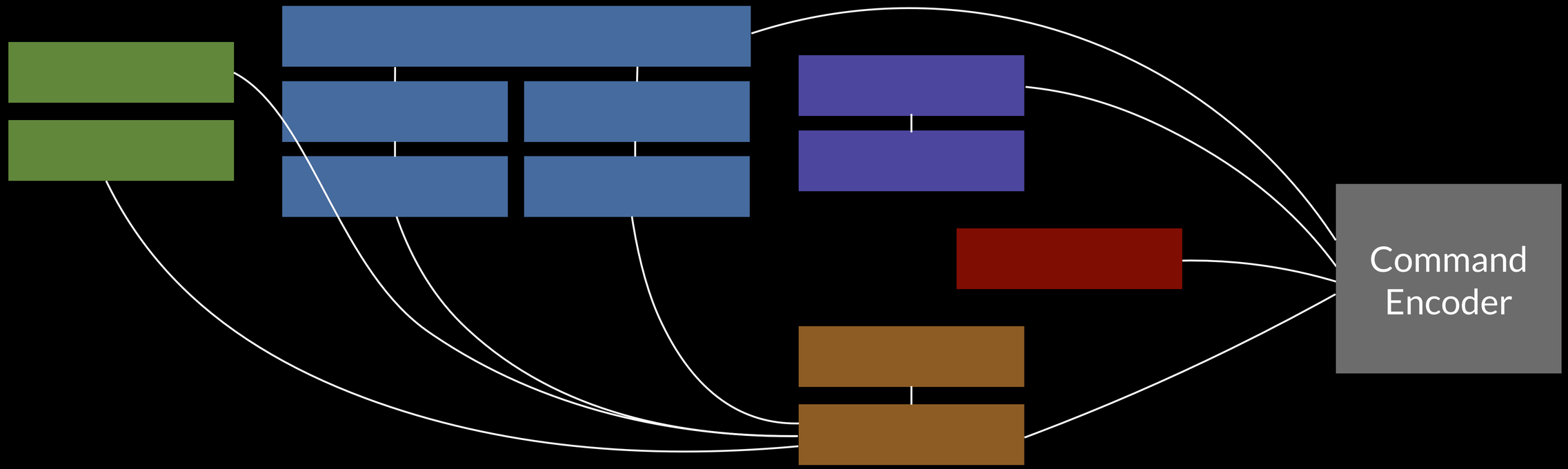


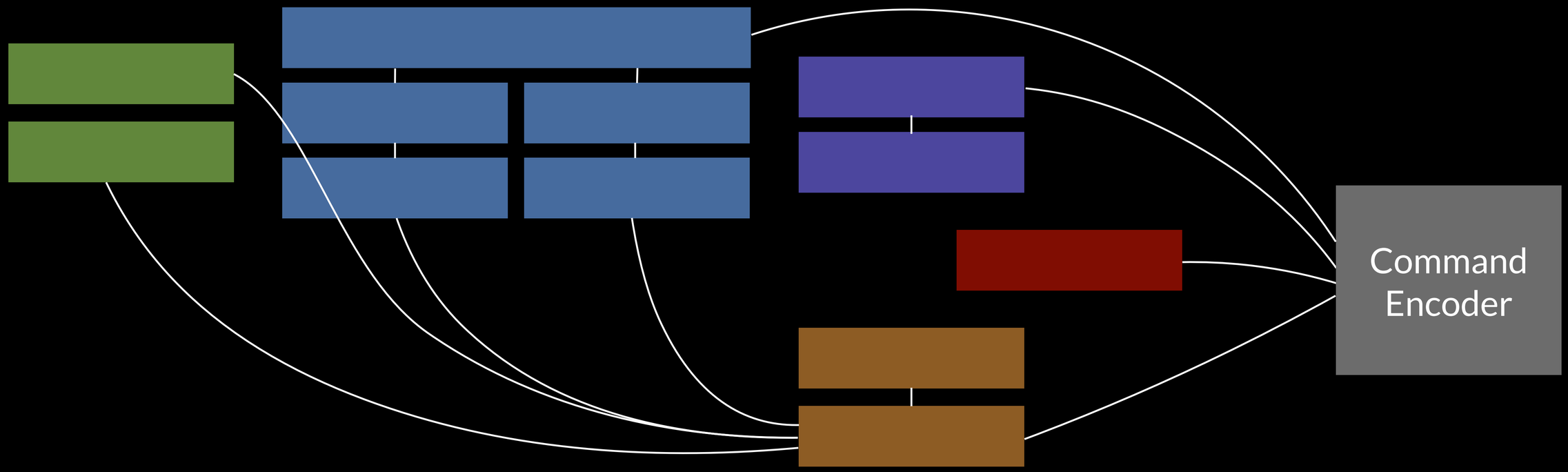


```
fn ( input: Texture, vertices: T[ ], output: Target )
```

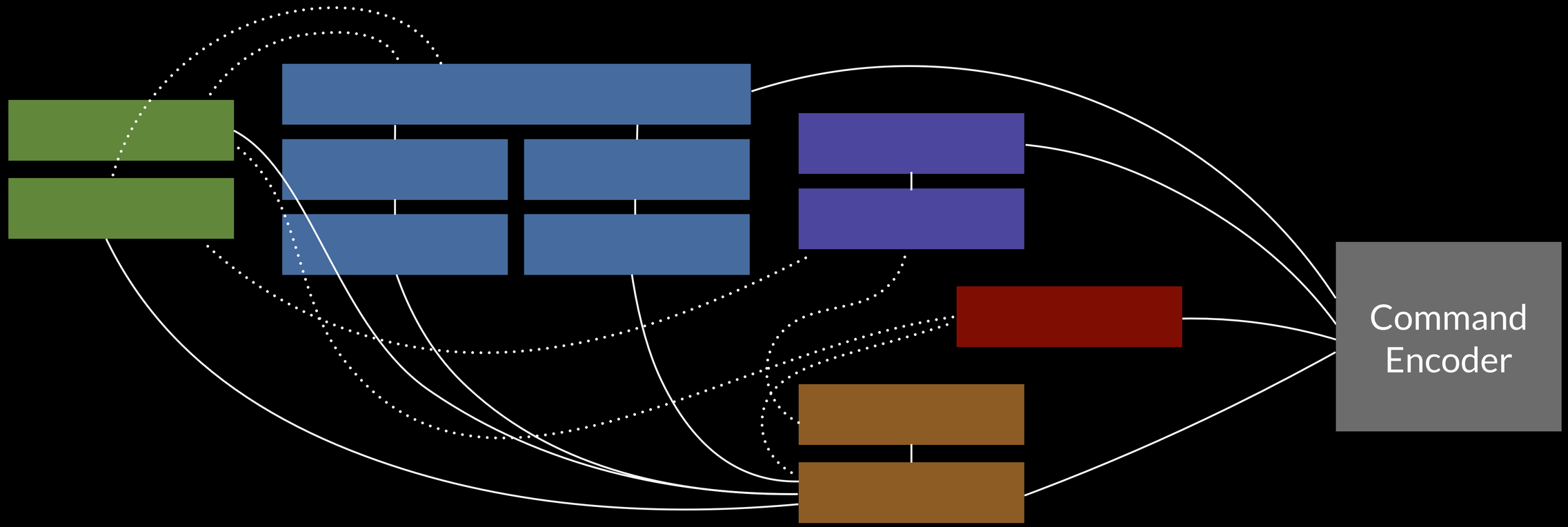





Command
Encoder

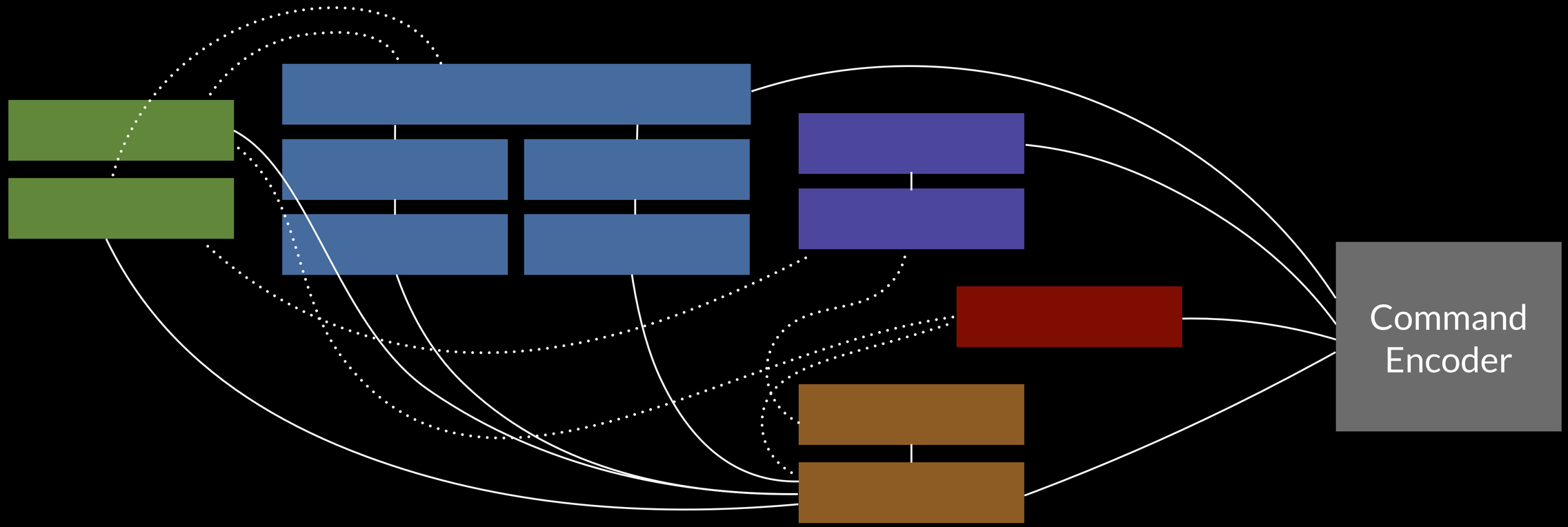


(Shader) => *(Pipeline)*
=> *(Targets, Uniforms, Vertices)*

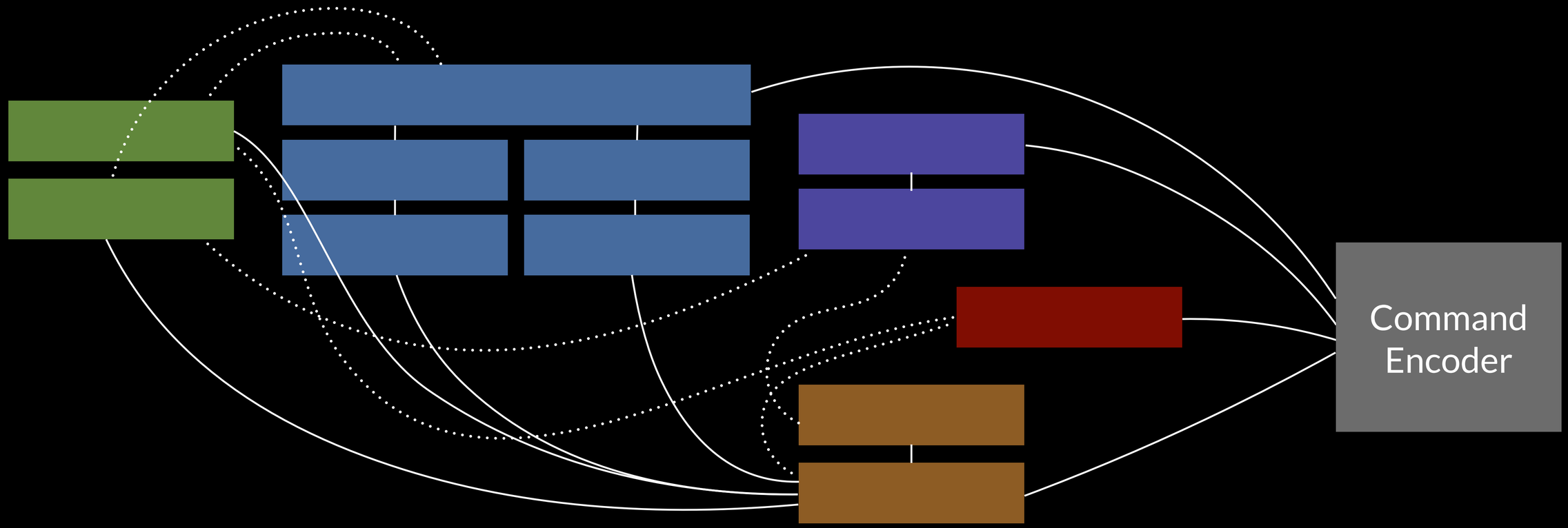


(Shader<T, U, V>) => (Pipeline<S, T, U, V>)
=> (Targets, Uniforms, Vertices)

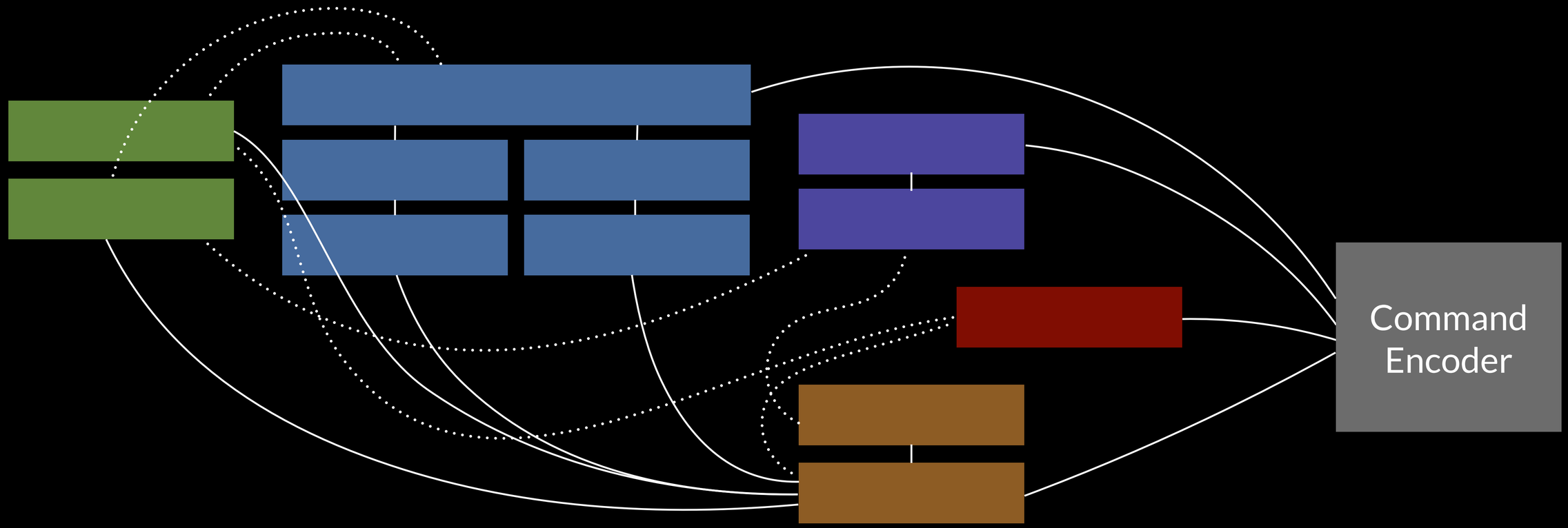
"How do I call a f😿ing function"



*(**S**hader<**T**, **U**, **V**>) => (Pipeline<**S**, **T**, **U**, **V**>)*
*=> (**T**argets, **U**niforms, **V**ertices)*

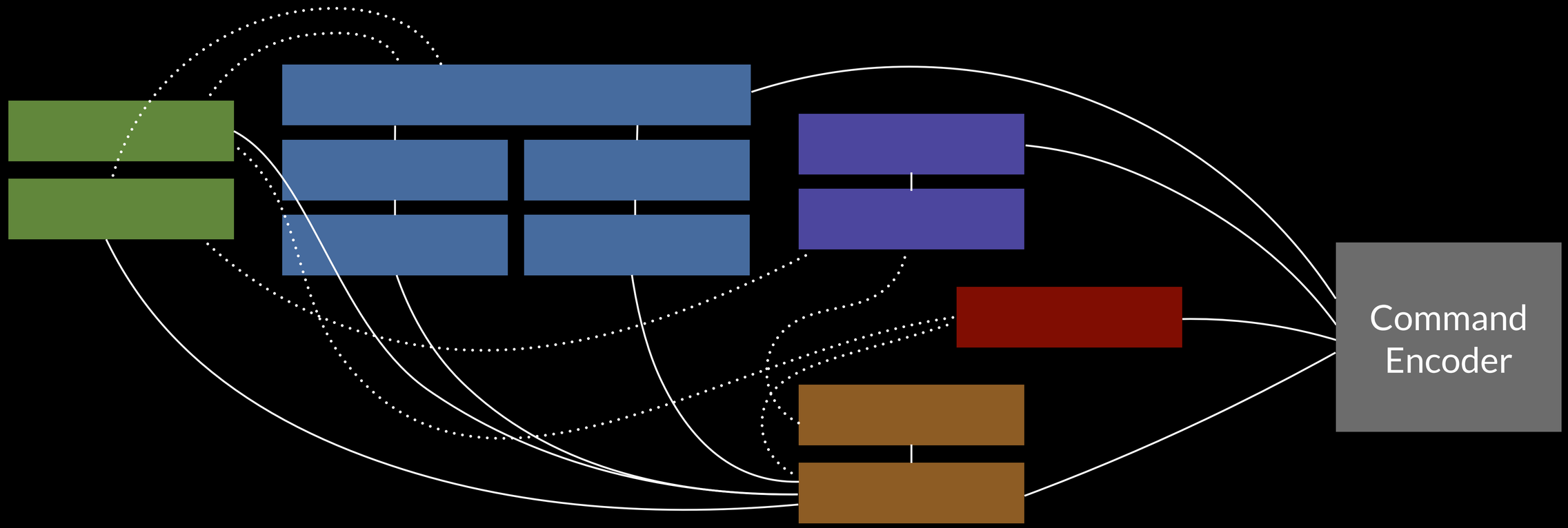


*(**S**hader<**T**, **U**, **V**>) => (Pipeline<**S**, **T**, **U**, **V**>)*
*=> (**T**argets, **U**niforms, **V**ertices)*



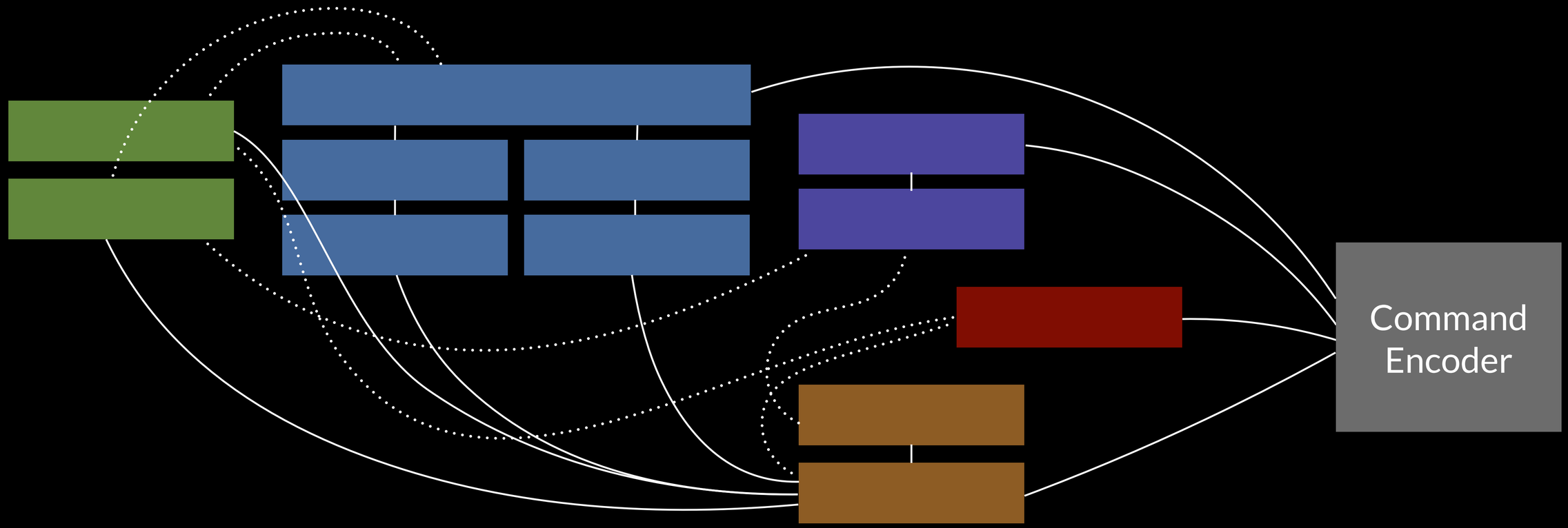
*(**S**hader<**T**, **U**, **V**>) => (Pipeline<**S**, **T**, **U**, **V**>)*
*=> (**T**argets, **U**niforms, **V**ertices)*

The calling convention is
a big nested data structure



*(Shader<**T**, **U**, **V**>) ↻ (Pipeline<**S**, **T**, **U**, **V**>)*
↻ (Targets ↻ Uniforms ↻ Vertices)

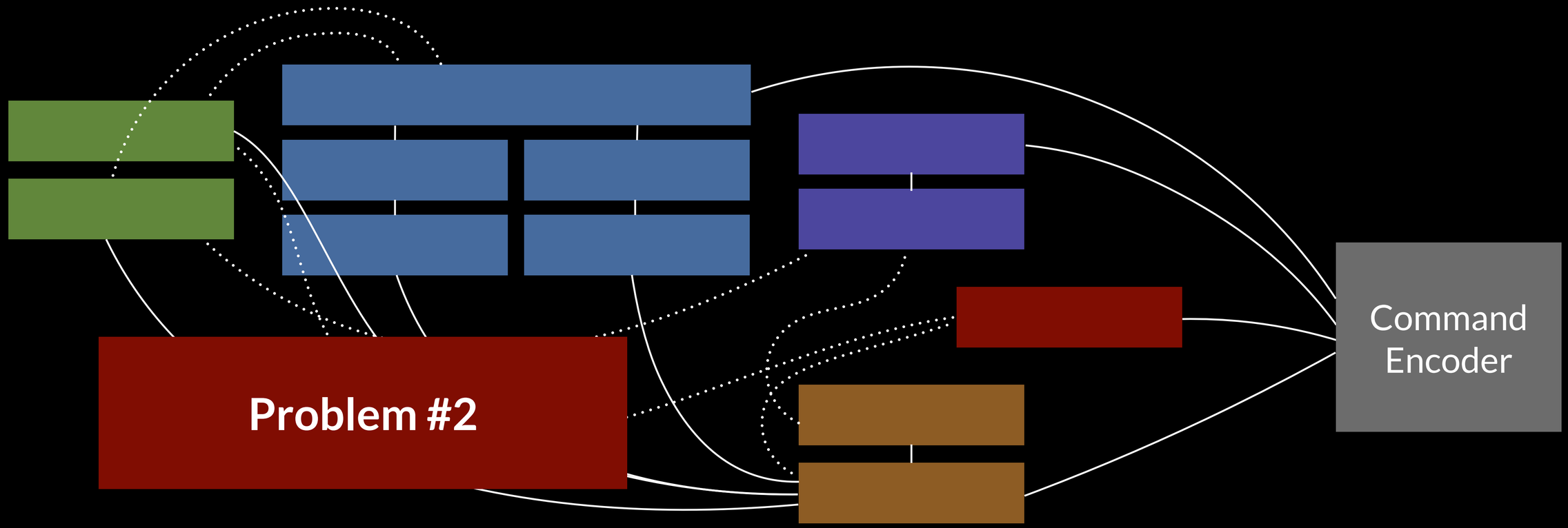
The calling convention is
a big nested data structure



*(Shader<**T**, **U**, **V**>) & (Pipeline<**S**, **T**, **U**, **V**>)*
& (Targets & Uniforms & Vertices)

The calling convention is
a big nested data structure

(*) Simplification
Reality is worse



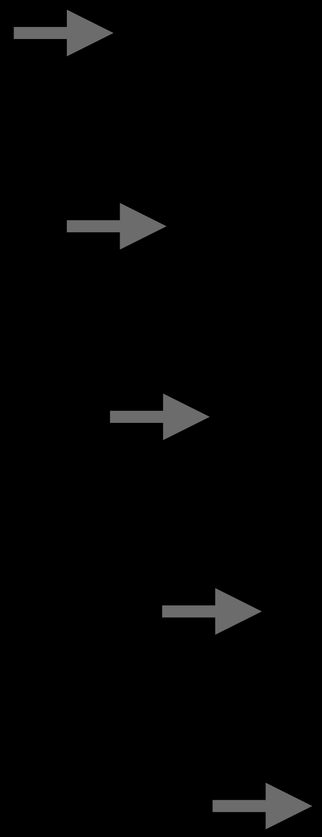
*(Shader<**T, U, V**>) ↻ (Pipeline<**S, T, U, V**>)*
↻ (Targets ↻ Uniforms ↻ Vertices)

The calling convention is
a big nested data structure

(*) Simplification
Reality is worse

Why???

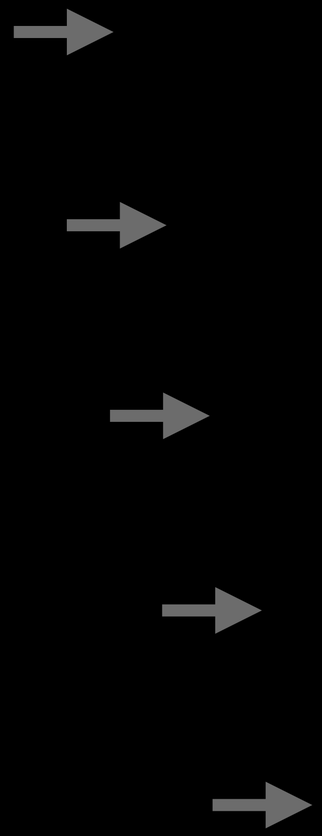
```
(Shader<T, U, V>) & (Pipeline<S, T, U, V>)
    & (Targets & Uniforms & Vertices)
```



Why???

*(Shader<**T**, **U**, **V**>) & (Pipeline<**S**, **T**, **U**, **V**>) & (Targets & Uniforms & Vertices)*

- A classic **video game renderer**:



Why???

*(Shader<**T**, **U**, **V**>) & (Pipeline<**S**, **T**, **U**, **V**>) & (Targets & Uniforms & Vertices)*

- A classic **video game renderer**:

```
for (purple of targets)
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      →
      →
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        →
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Everything set up
and **grouped ahead of time**

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Everything set up
and **grouped ahead of time**

Inner loops change state
more granularly
than outer loops

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Everything set up
and **grouped ahead of time**

Inner loops change state
more granularly
than outer loops

API is over-optimized
for **1 way** of doing
retained mode 3D graphics

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

4 bind groups max

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple); setUniforms(…)
  for (orange of pipelines[purple])
    setPipeline(orange); setUniforms(…)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red); setUniforms(…)
        draw( ... )
```

Everything set up
and **grouped ahead of time**

Inner loops change state
more granularly
than outer loops

API is over-optimized
for **1 way** of doing
retained mode 3D graphics

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple); setUniforms(…)
  for (orange of [illegible][purple])
    setPip[illegible]forms(…)
    for ([illegible]e])
      setUni[illegible]
      for (red of objects[blu[illegible]])
        setVertices(red); setUniforms(…)
        draw( … )
```

But I want to **compose functions**

4 bind groups max

Everything set up and **grouped ahead of time**

Inner loops change state **more granularly** than outer loops

API is over-optimized for **1 way** of doing **retained mode** 3D graphics

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

Big issue with the **setup**

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Big issue with the **setup**

If the **size** of the data changes, you need **new storage** and **upload**

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Big issue with the **setup**

If the **size** of the data changes, you need **new storage** and **upload**

If the **type** of the data changes, you need **new everything**

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Track all the **resources** and **cache** them

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Track all the **resources** and **cache** them

Deal with **async**

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Track all the **resources** and **cache** them

Deal with **async**

Pack all your data in **binary arrays**

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Track all the **resources** and **cache** them

Deal with **async**

Pack all your data in **binary arrays**

But I want to **draw lines**

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Pre-process and **bake data**

Hand-tune performance

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Pre-process and **bake data**

Hand-tune performance

Fork your tech stack

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Pre-process and **bake data**

Hand-tune performance

Fork your tech stack

Fix all the new **bugs**

Why???

(Shader<T, U, V>) & (Pipeline<S, T, U, V>) & (Targets & Uniforms & Vertices)

- A classic **video game renderer**:

```
for (purple of targets)
  setTarget(purple)
  for (orange of pipelines[purple])
    setPipeline(orange)
    for (blue of materials[orange])
      setUniforms(blue)
      for (red of objects[blue])
        setVertices(red)
        draw( ... )
```

Pre-process and **bake data**

Hand-tune performance

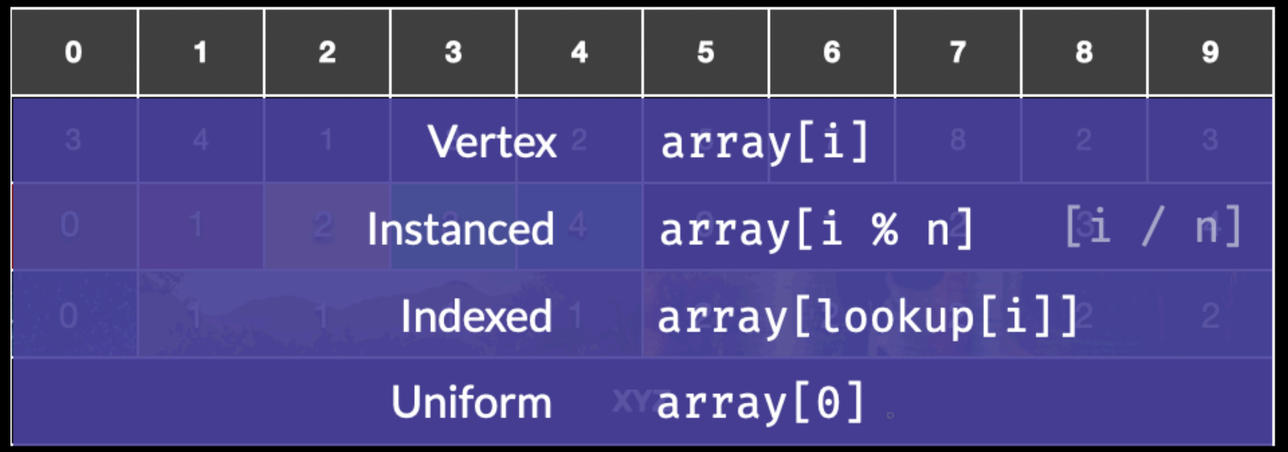
Fork your tech stack

But I don't have **time**

Fix all the new **bugs**

Spread Policies

Problem #1



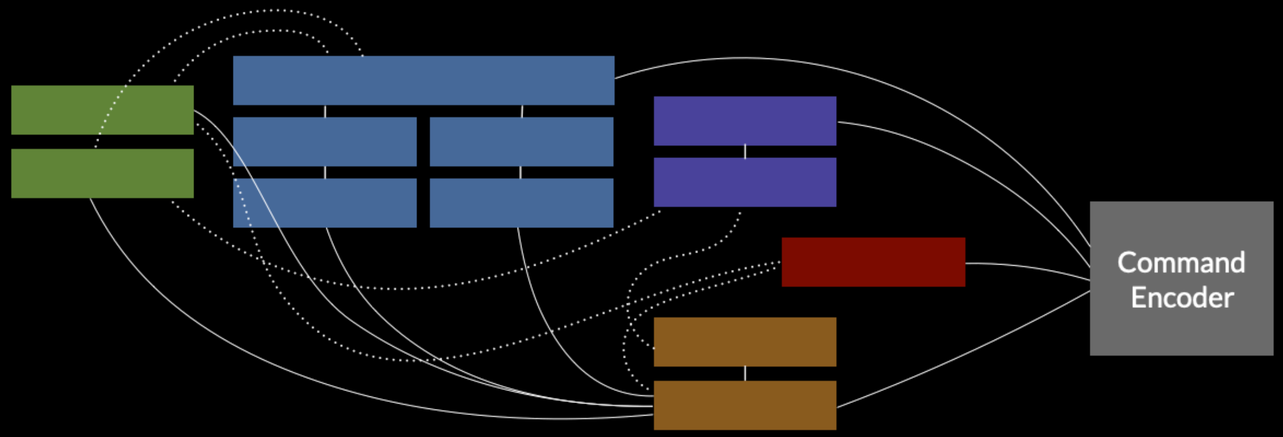
Spread Policies

Problem #1

0	1	2	3	4	5	6	7	8	9
Vertex					array[i]				
Instanced					array[i % n]			[i / n]	
Indexed					array[lookup[i]]				
Uniform					array[0]				

Pipeline Dispatch

Problem #2



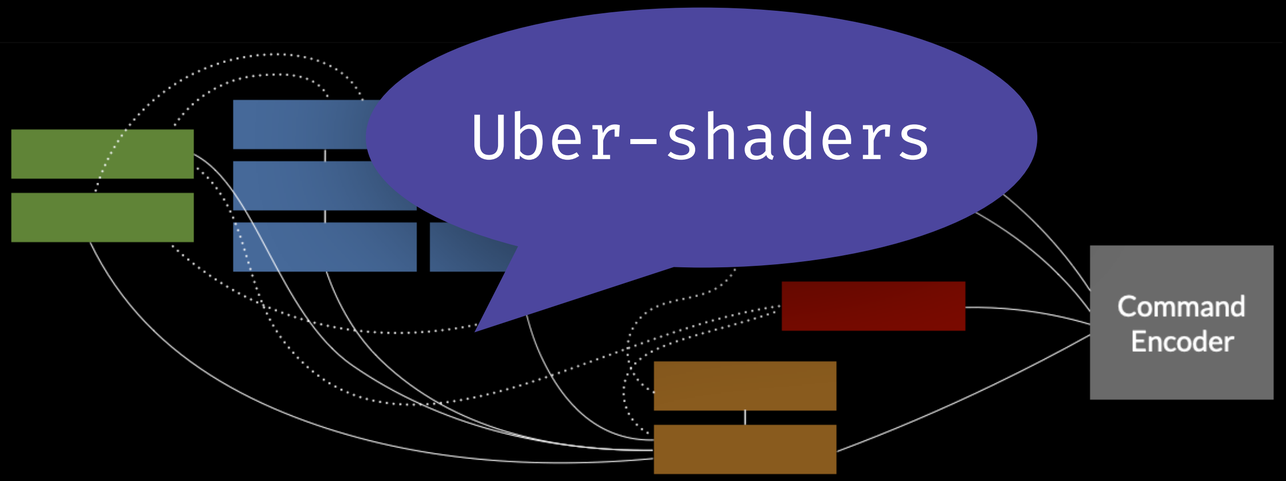
Spread Policies

Problem #1

0	1	2	3	4	5	6	7	8	9
Vertex					array[i]				
Instanced					array[i % n]			[i / n]	
Indexed					array[lookup[i]]				
Uniform					array[0]				

Pipeline Dispatch

Problem #2



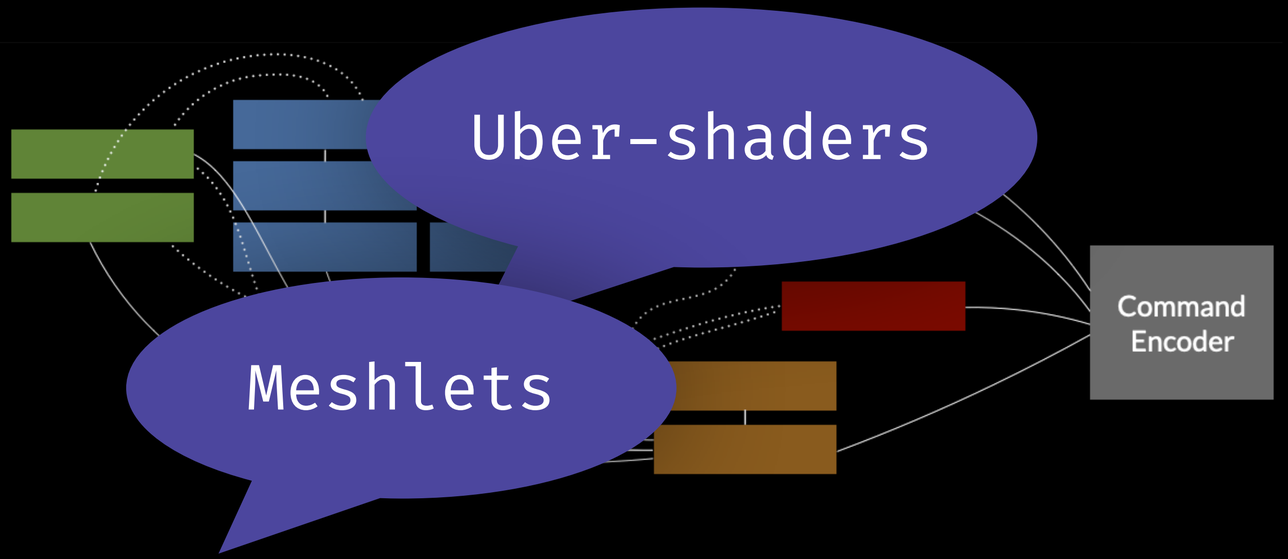
Spread Policies

Problem #1

0	1	2	3	4	5	6	7	8	9
Vertex					array[i]				
Instanced					array[i % n]			[i / n]	
Indexed					array[lookup[i]]				
Uniform					array[0]				

Pipeline Dispatch

Problem #2



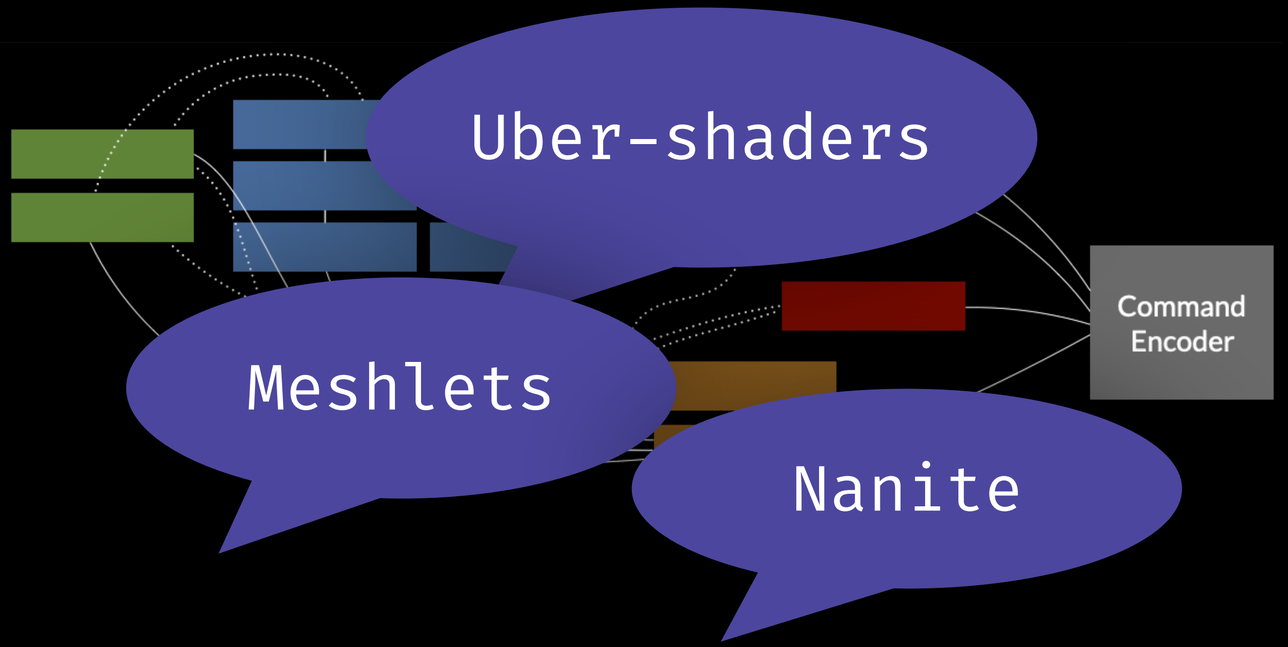
Spread Policies

Problem #1

0	1	2	3	4	5	6	7	8	9
Vertex					array[i]				
Instanced					array[i % n]			[i / n]	
Indexed					array[lookup[i]]				
Uniform					array[0]				

Pipeline Dispatch

Problem #2



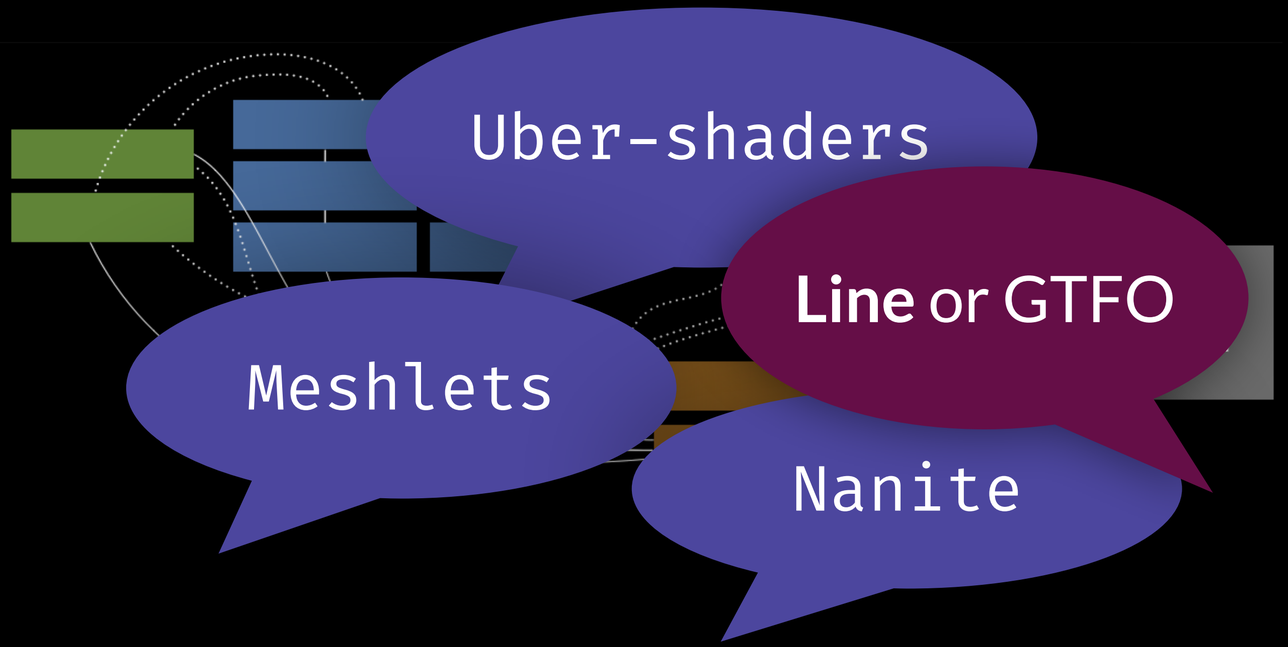
Spread Policies

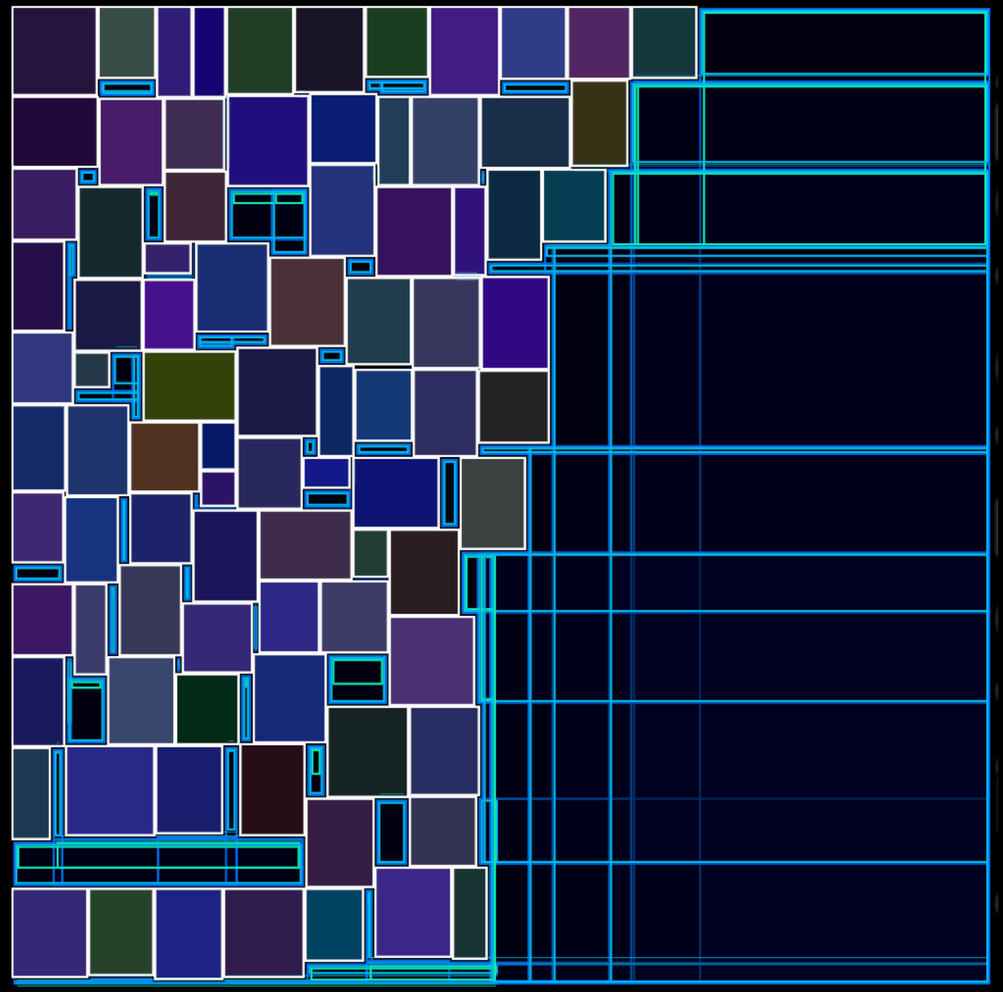
Problem #1

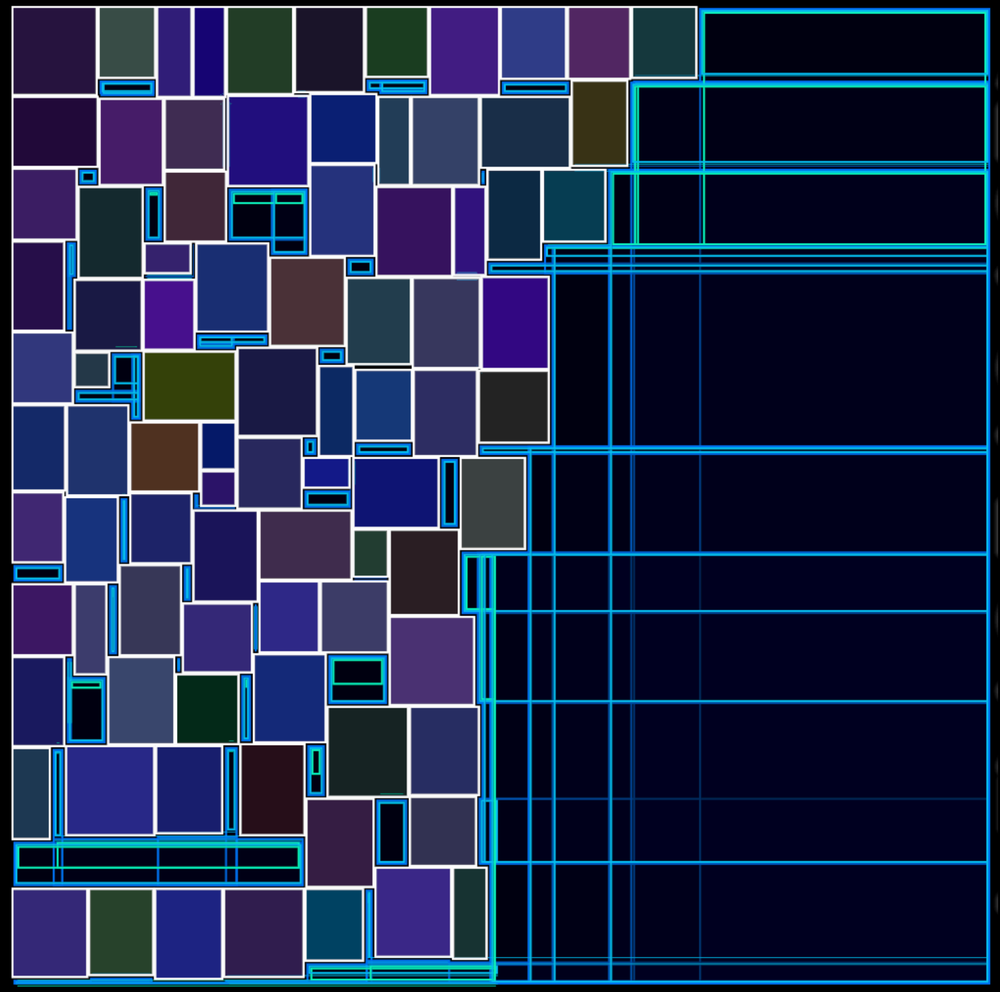
0	1	2	3	4	5	6	7	8	9
Vertex					array[i]				
Instanced					array[i % n]			[i / n]	
Indexed					array[lookup[i]]				
Uniform					array[0]				

Pipeline Dispatch

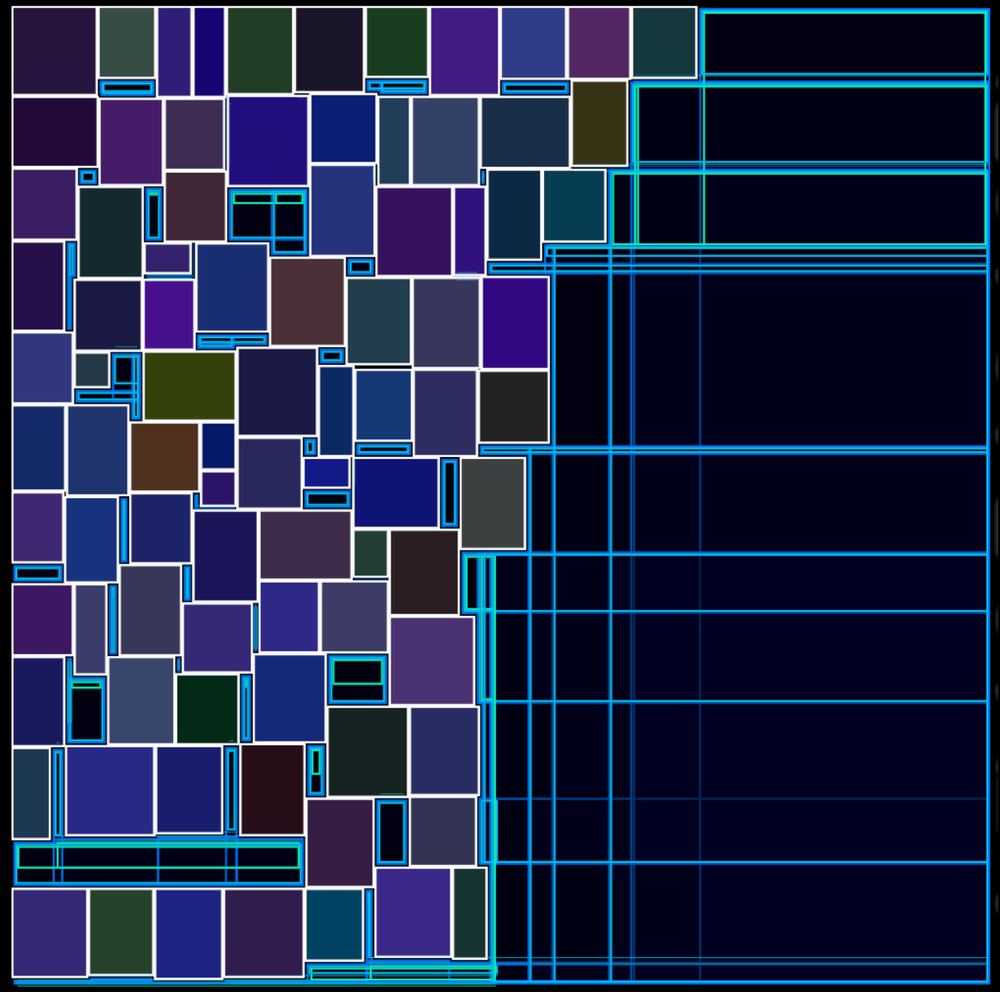
Problem #2





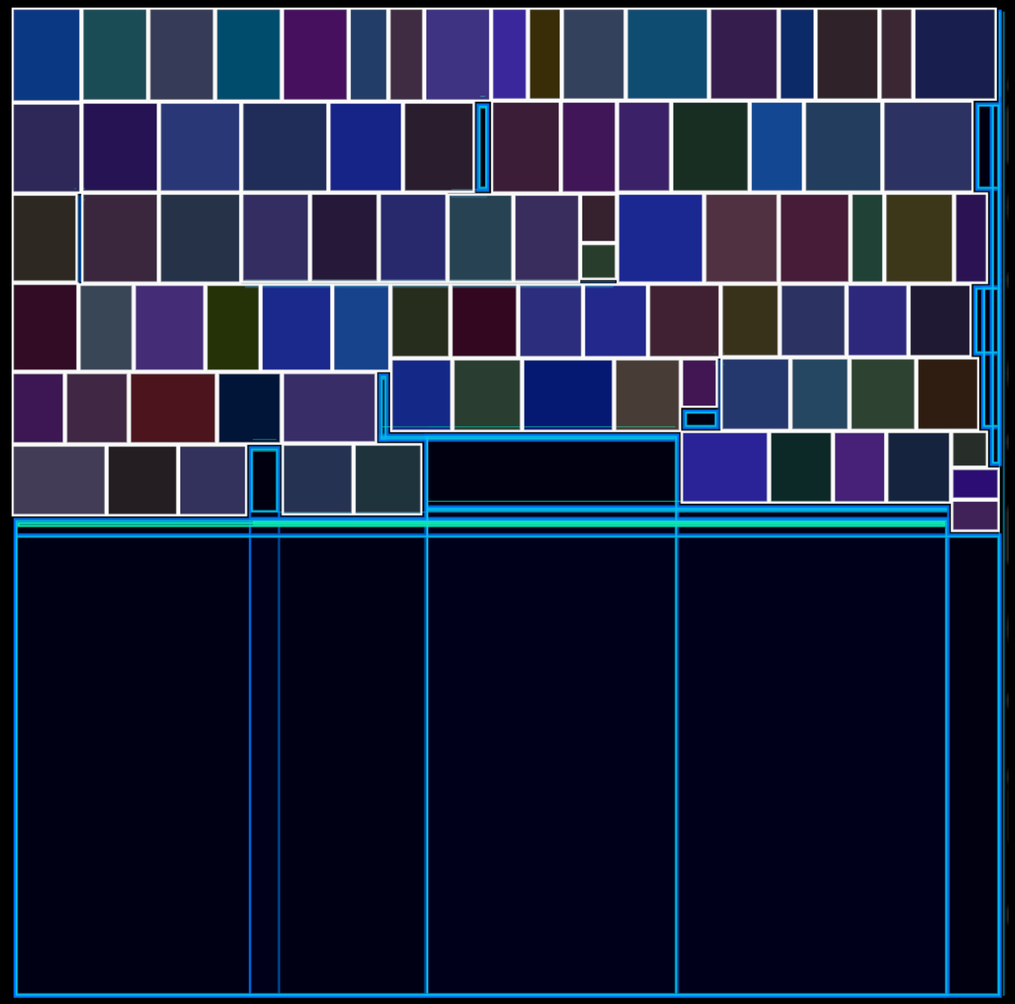


An atlas is just a
spicy memory allocator



An atlas is just a
spicy memory allocator

Retained mode
is the **solution** that causes
the **problem**



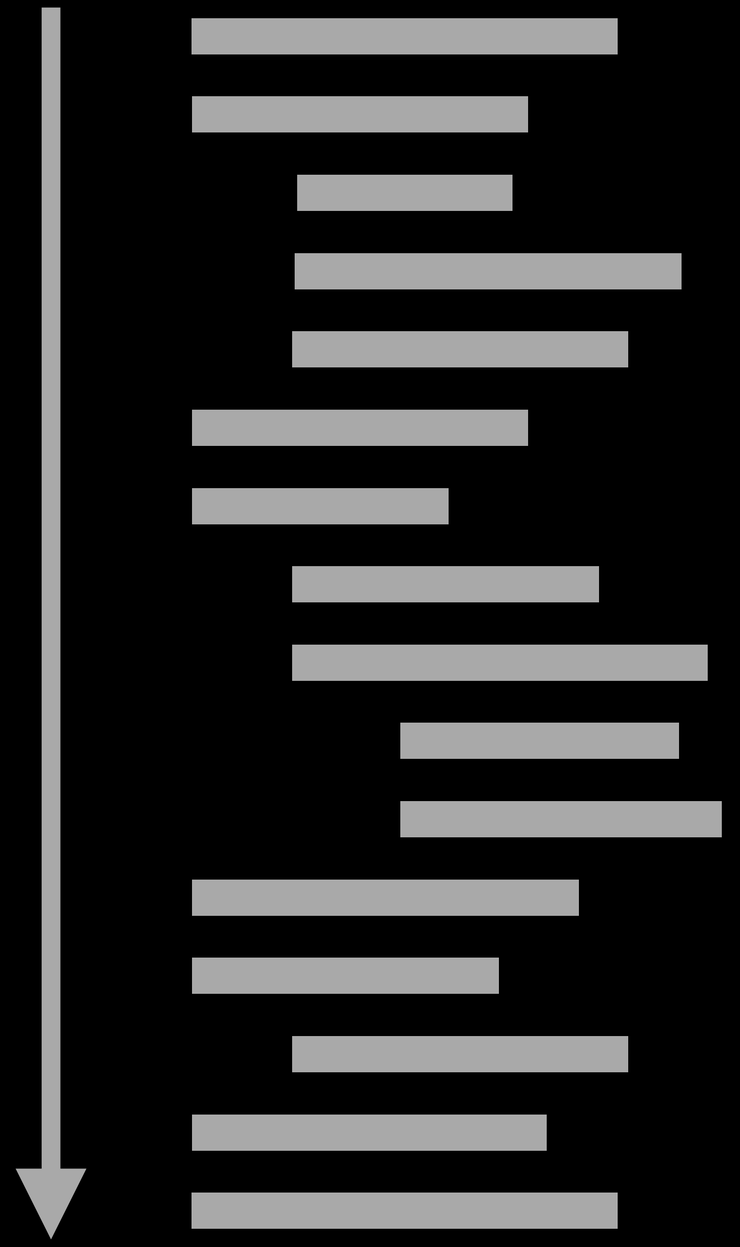
An atlas is just a
spicy memory allocator

Retained mode
is the **solution** that causes
the **problem**

Can't order blocks from
large to small

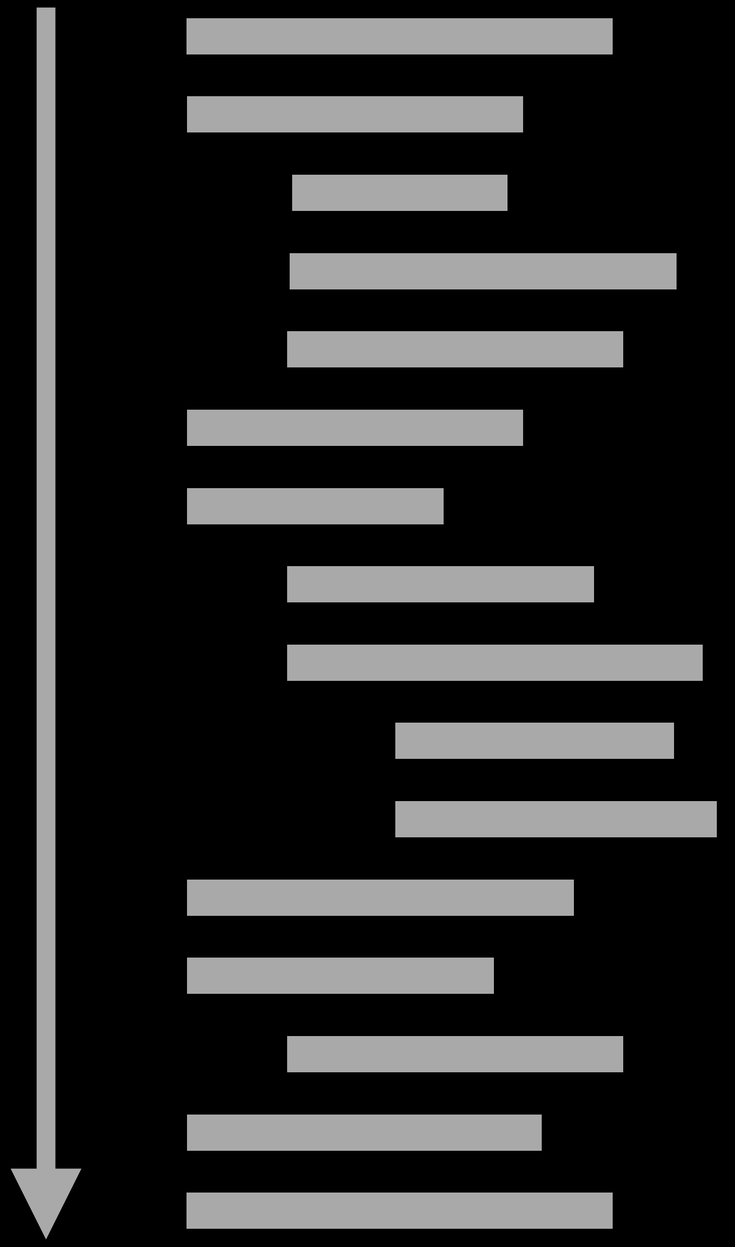
Live run-time

Immediate Mode



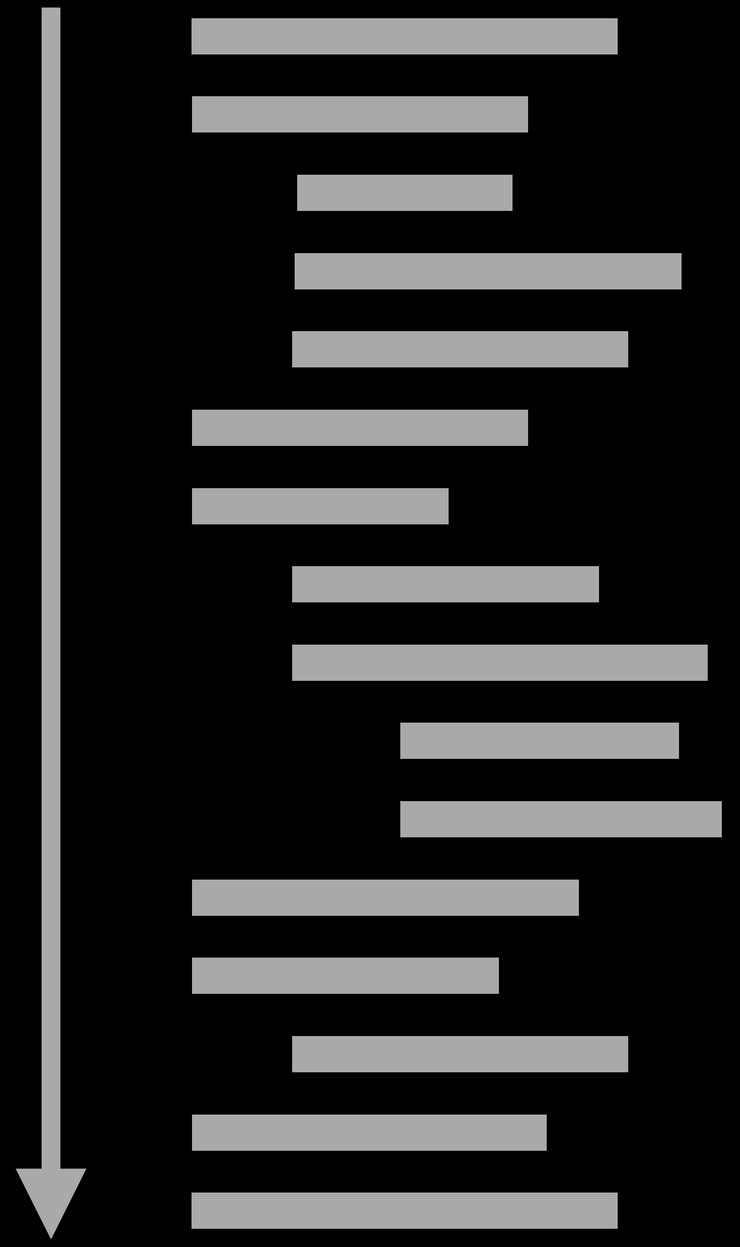
Immediate Mode

- Write all the code to produce 1 frame



Immediate Mode

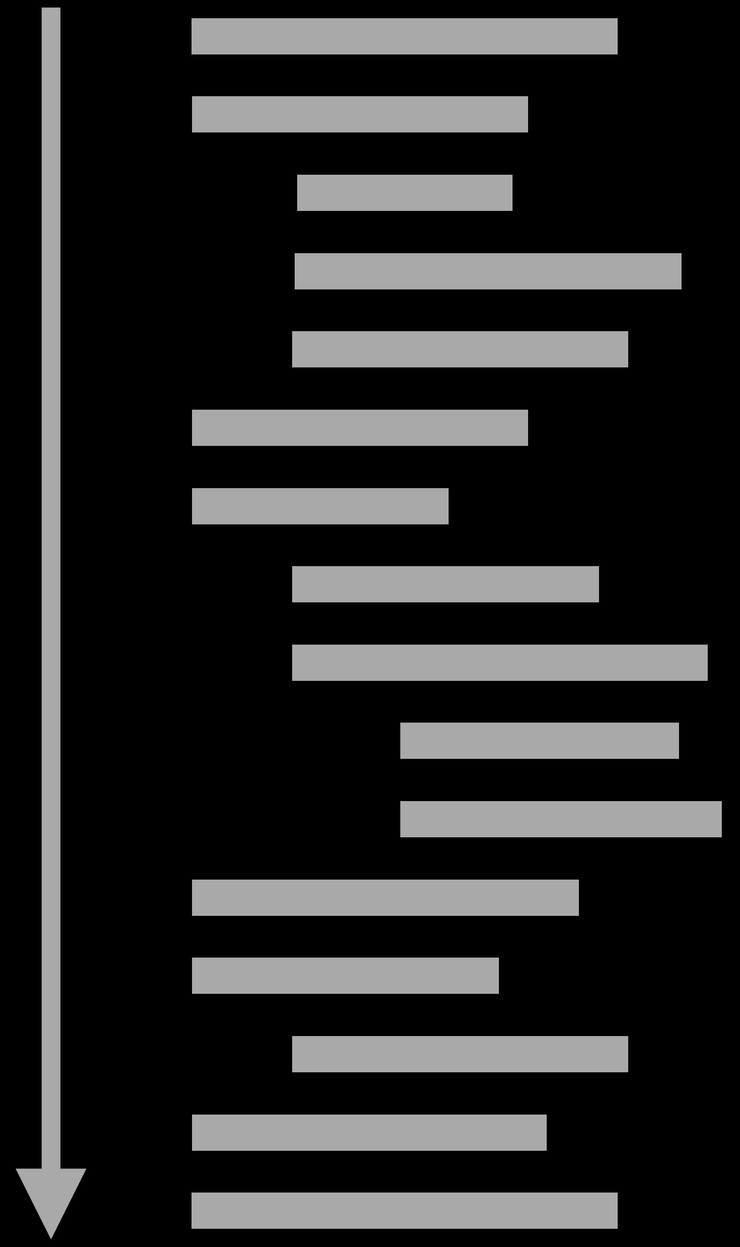
- Write all the code to produce 1 frame
- Call it again with new input



Immediate Mode

- Write all the code to produce 1 frame
- Call it again with new input

✓ Very simple

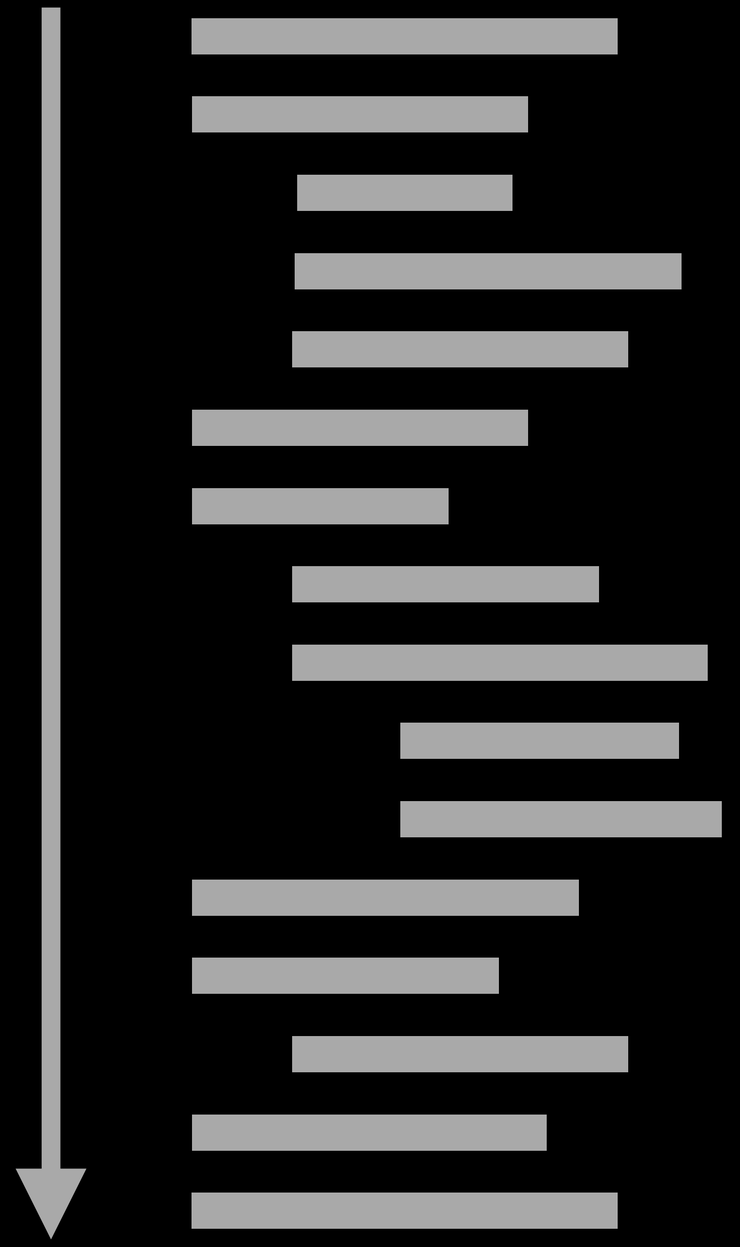


Immediate Mode

- Write all the code to produce 1 frame
- Call it again with new input

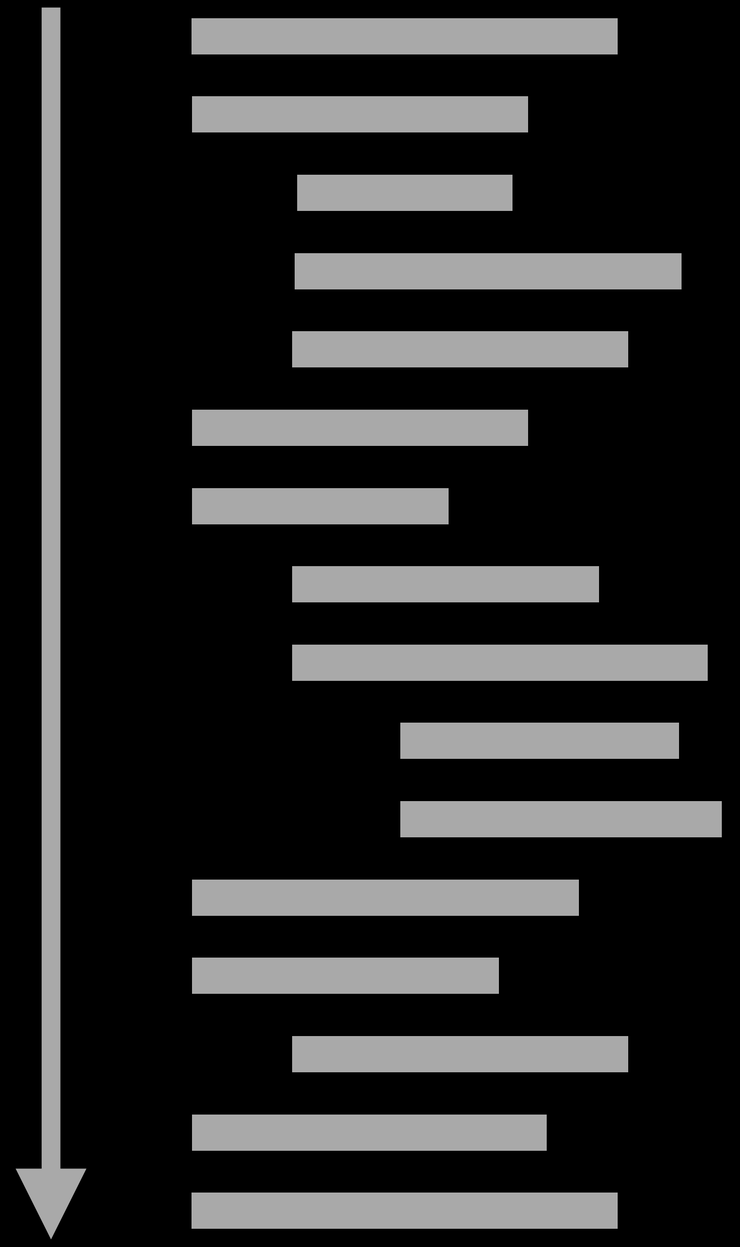
✓ Very simple

✗ Doesn't scale



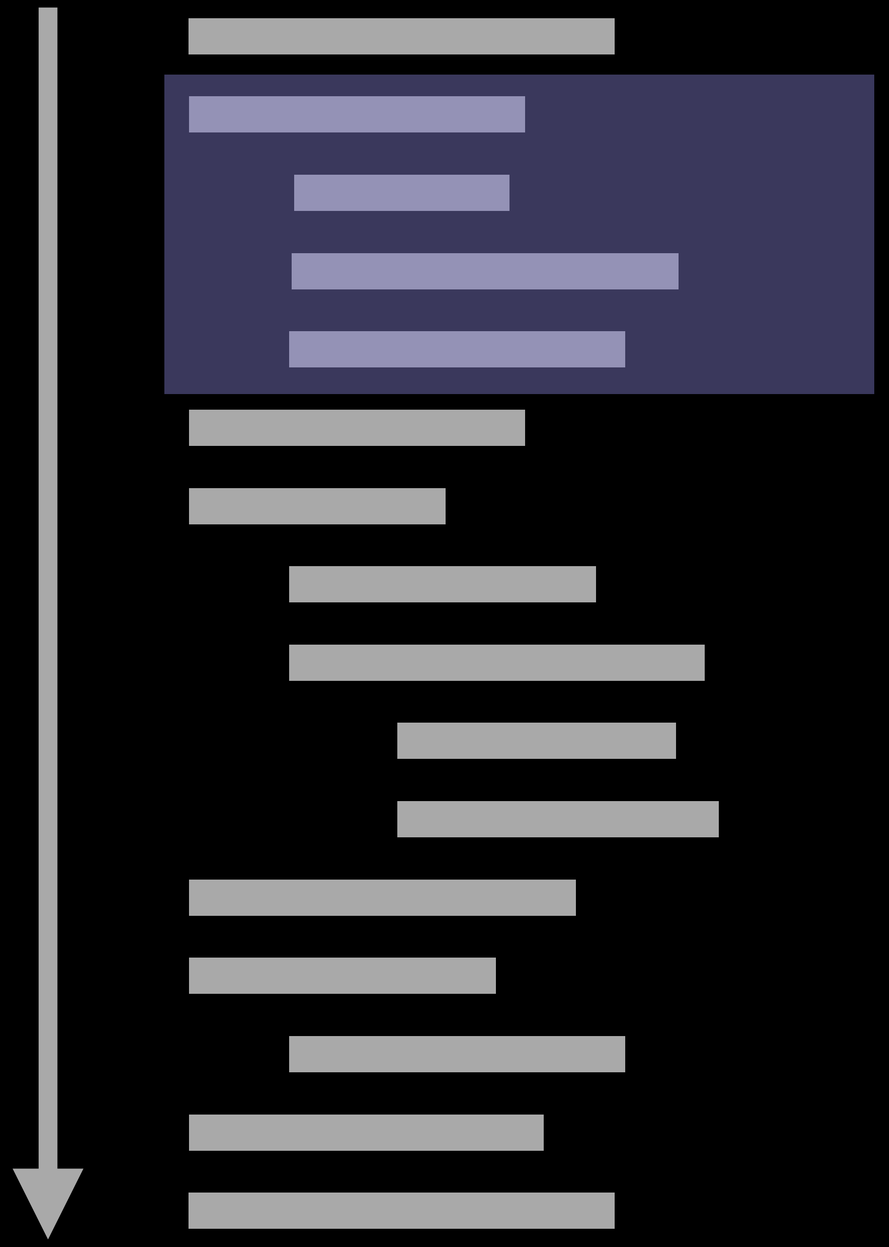
Incremental Mode

- Write all the code to produce 1 frame
- Memoize the slow parts
  = *Save last input and output*



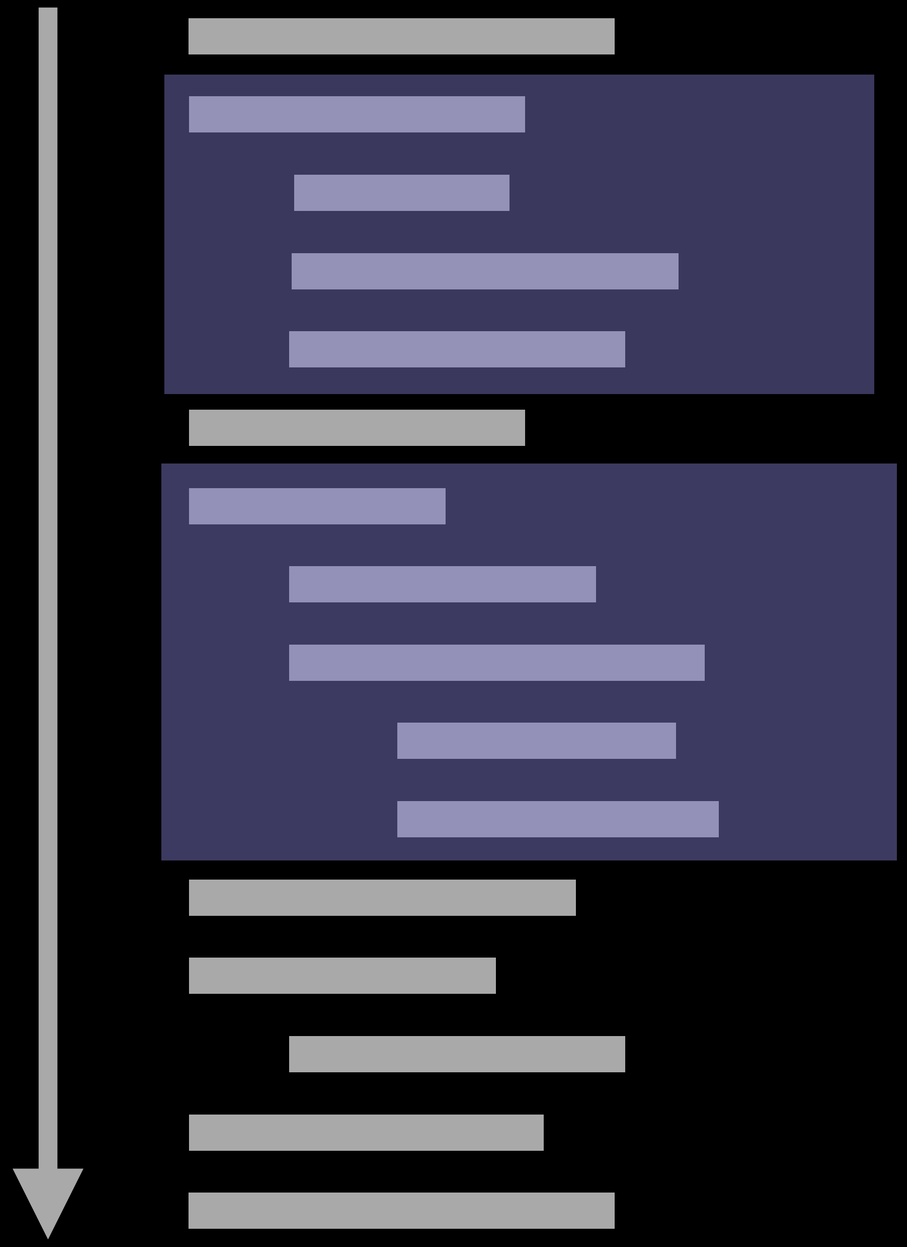
Incremental Mode

- Write all the code to produce 1 frame
- Memoize the slow parts
  = *Save last input and output*



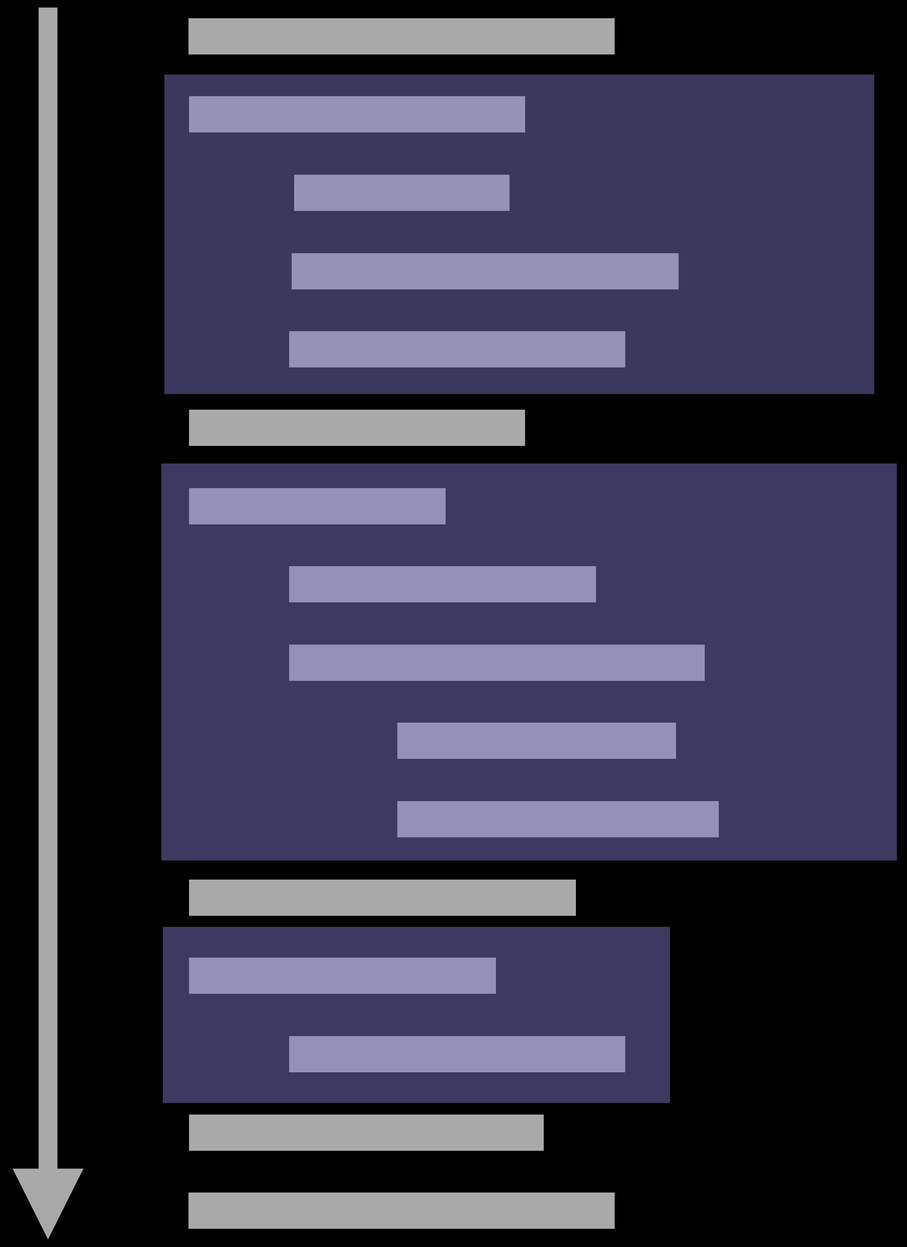
Incremental Mode

- Write all the code to produce 1 frame
- Memoize the slow parts
  = *Save last input and output*



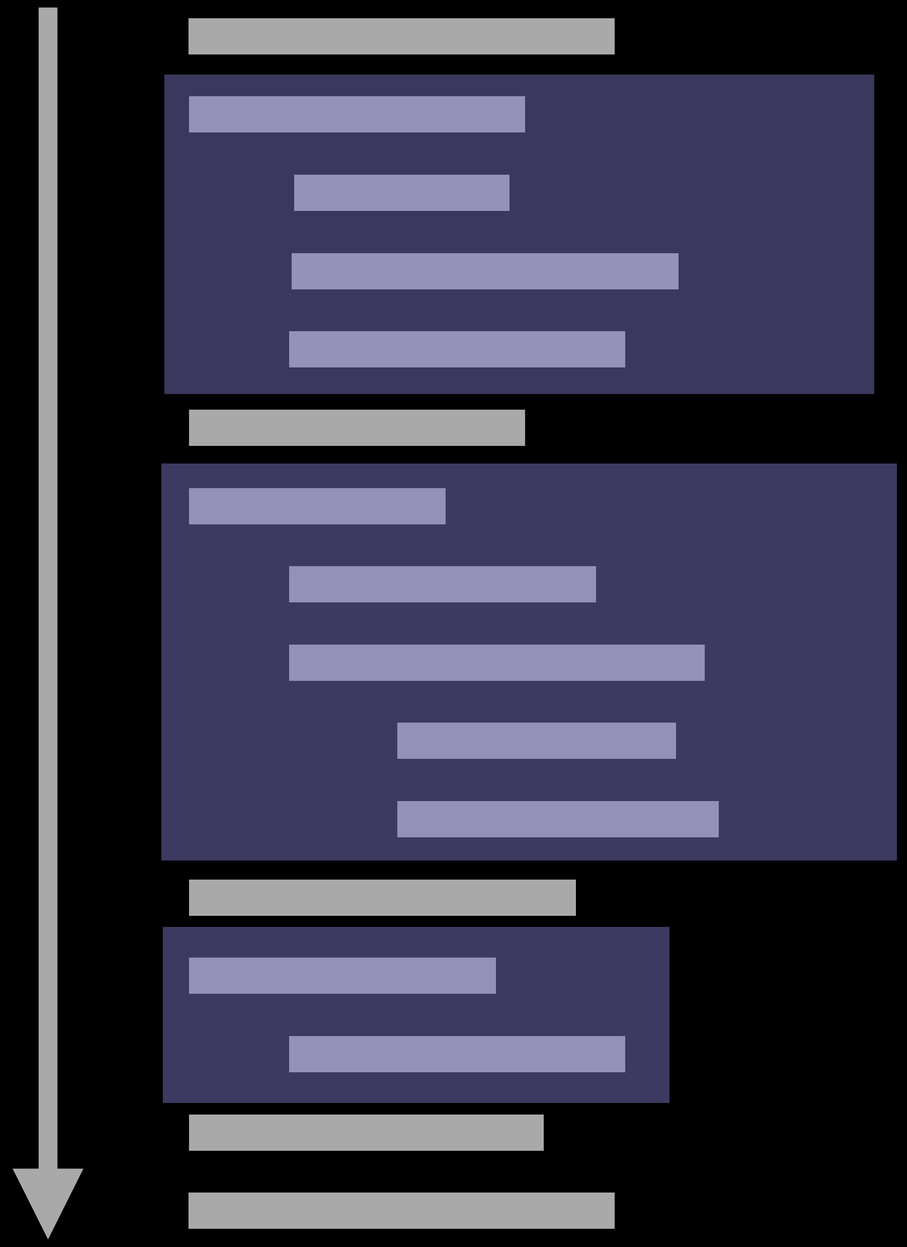
Incremental Mode

- Write all the code to produce 1 frame
- Memoize the slow parts
  = *Save last input and output*



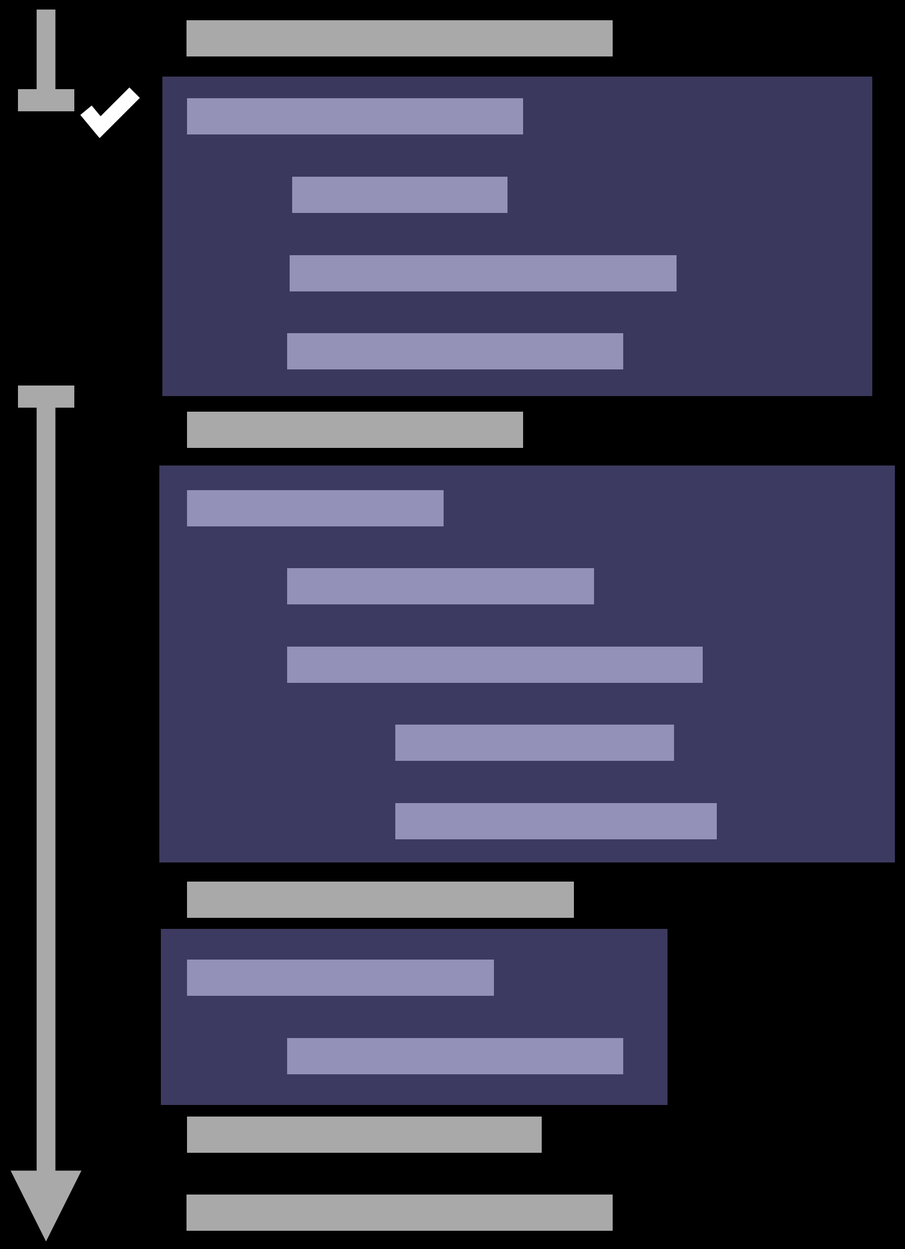
Incremental Mode

- Write all the code to produce 1 frame
- Memoize the slow parts
  = *Save last input and output*
- Call it repeatedly with new input



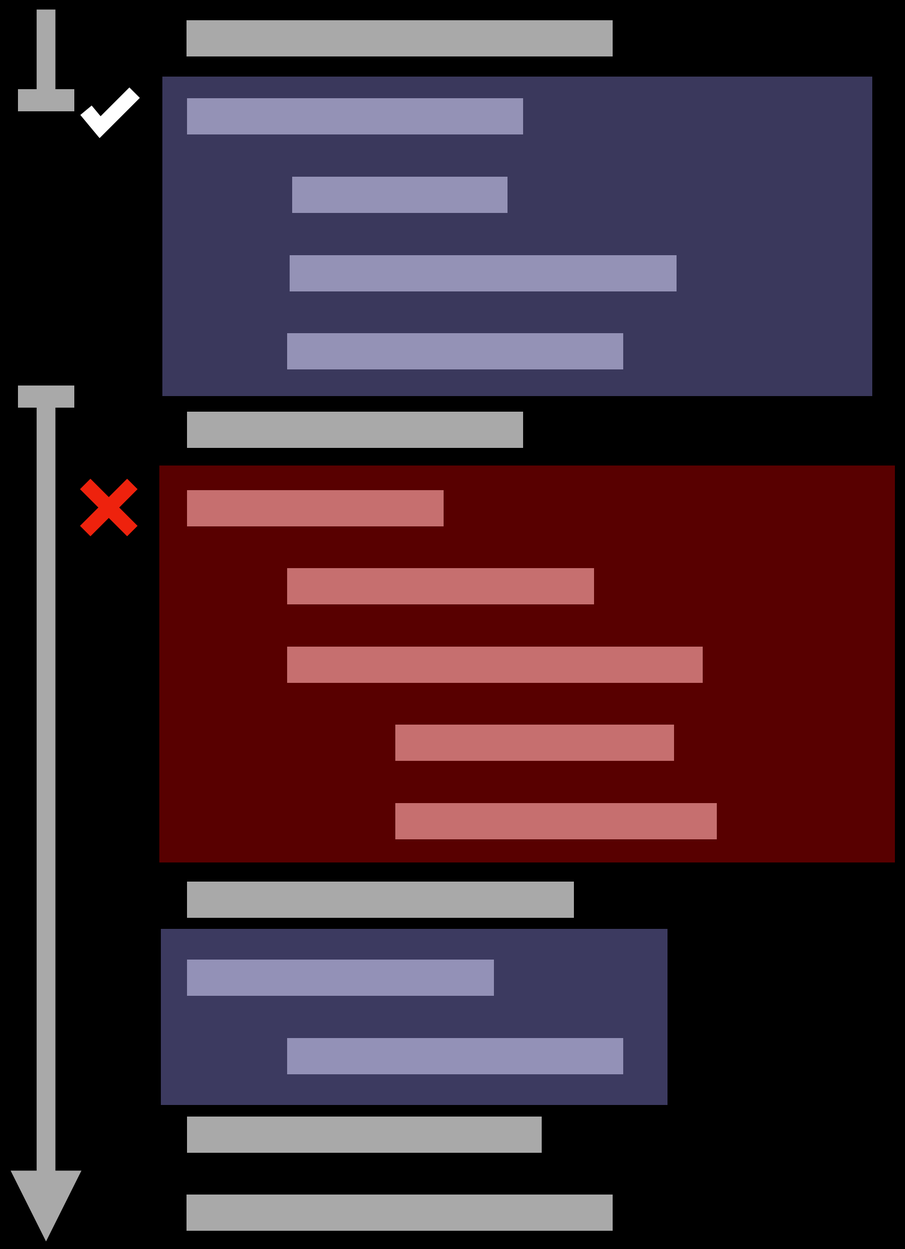
Incremental Mode

- Write all the code to produce 1 frame
- Memoize the slow parts
  = *Save last input and output*
- Call it repeatedly with new input



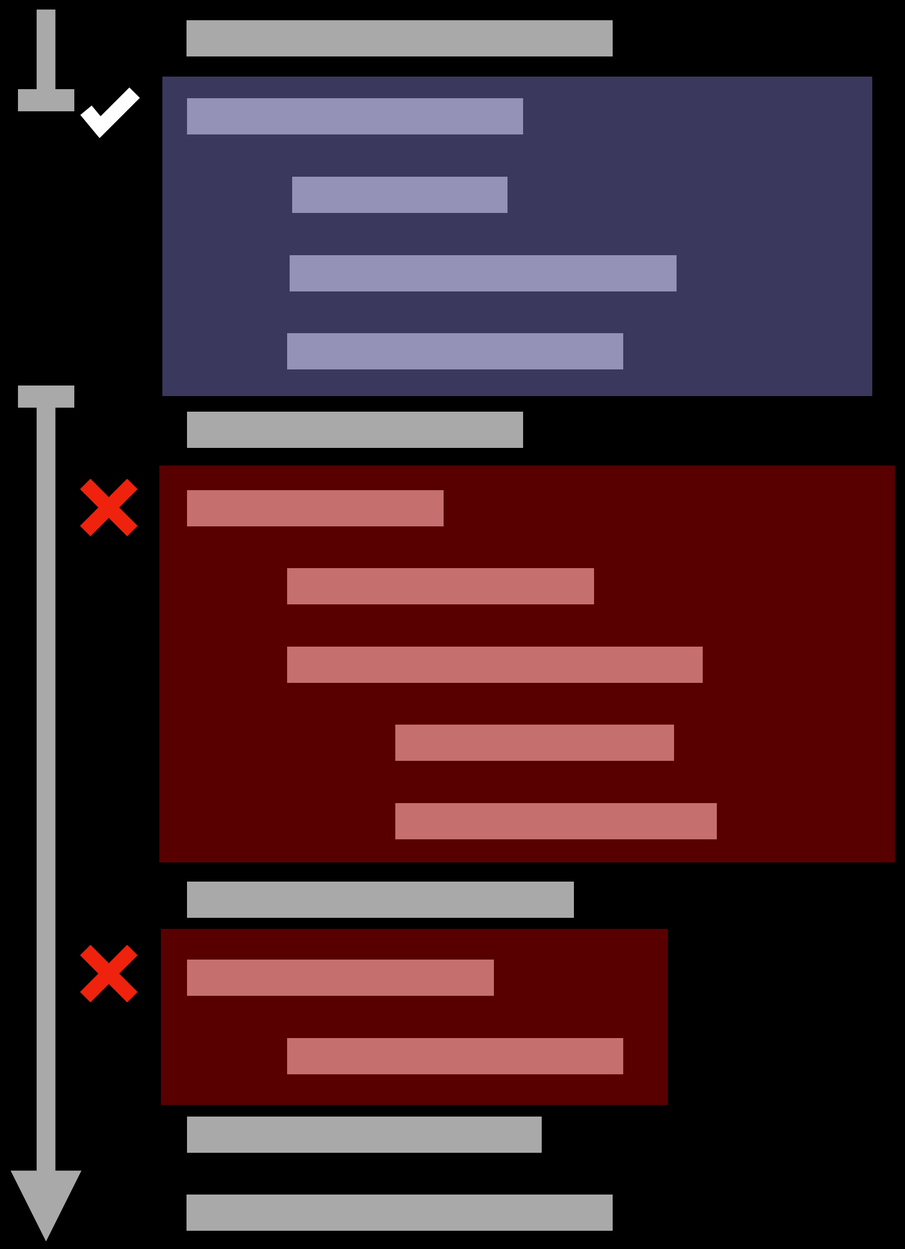
Incremental Mode

- Write all the code to produce 1 frame
- Memoize the slow parts
  = *Save last input and output*
- Call it repeatedly with new input



Incremental Mode

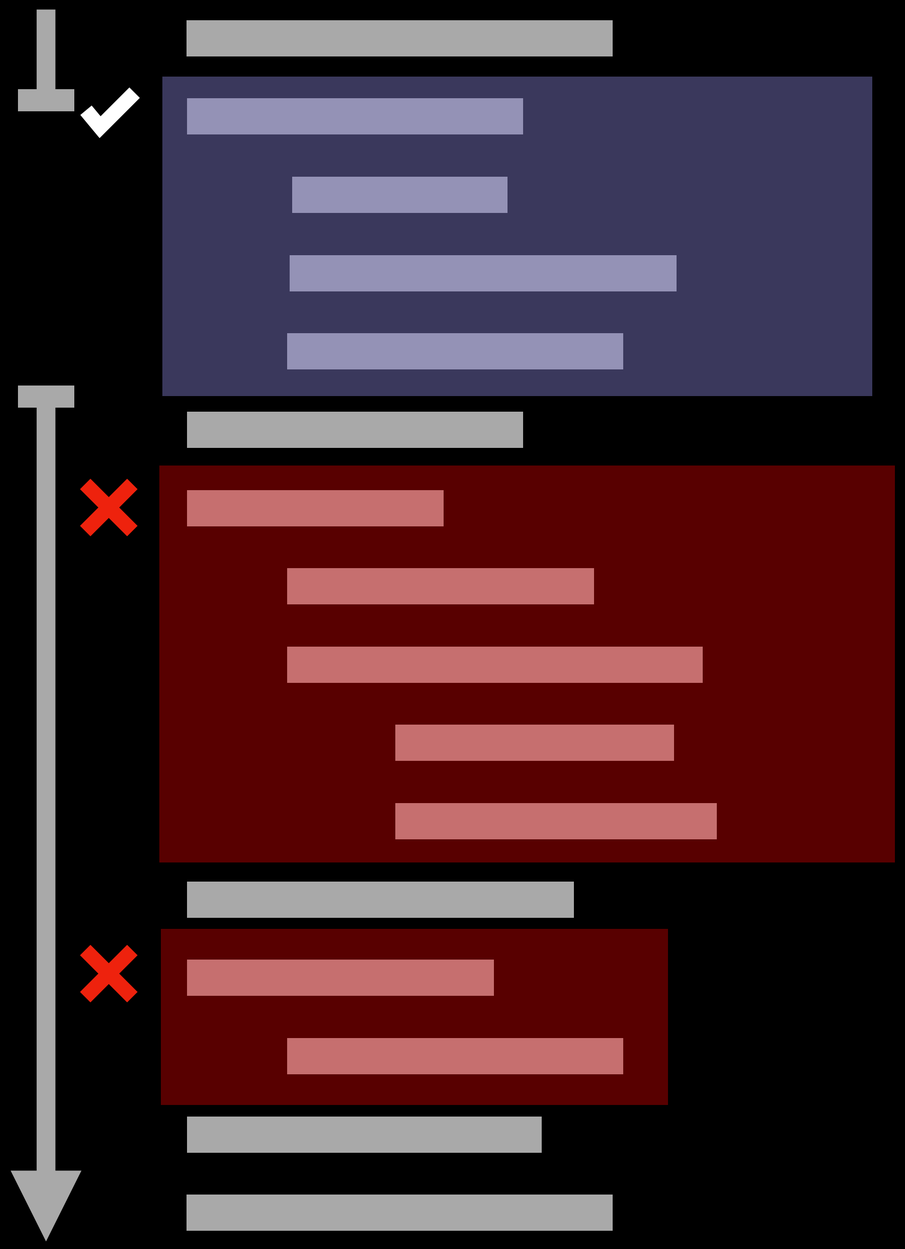
- Write all the code to produce 1 frame
- Memoize the slow parts
  = *Save last input and output*
- Call it repeatedly with new input



Incremental Mode

- Write all the code to produce 1 frame
- Memoize the slow parts
  = *Save last input and output*
- Call it repeatedly with new input

✓ Simple

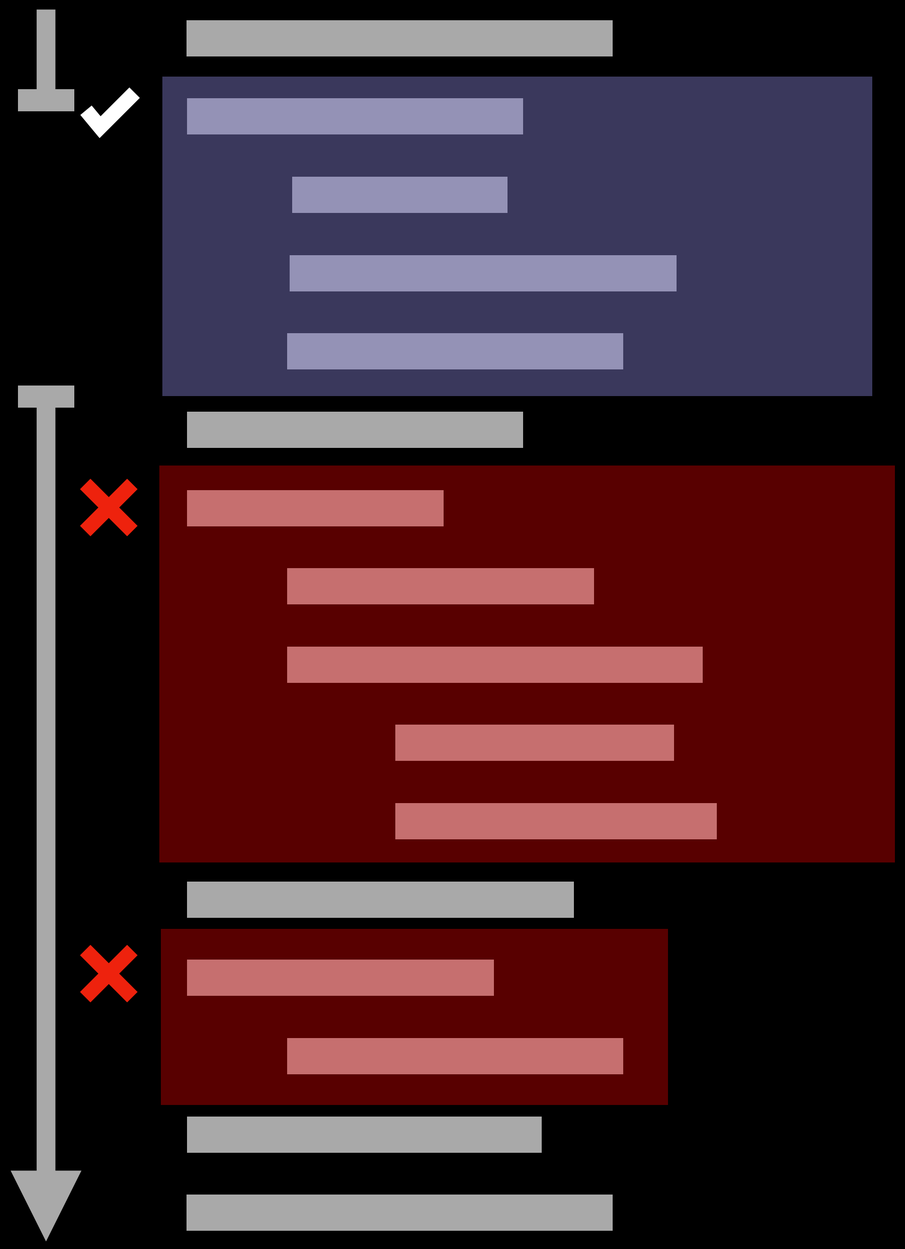


Incremental Mode

- Write all the code to produce 1 frame
- Memoize the slow parts
  = *Save last input and output*
- Call it repeatedly with new input

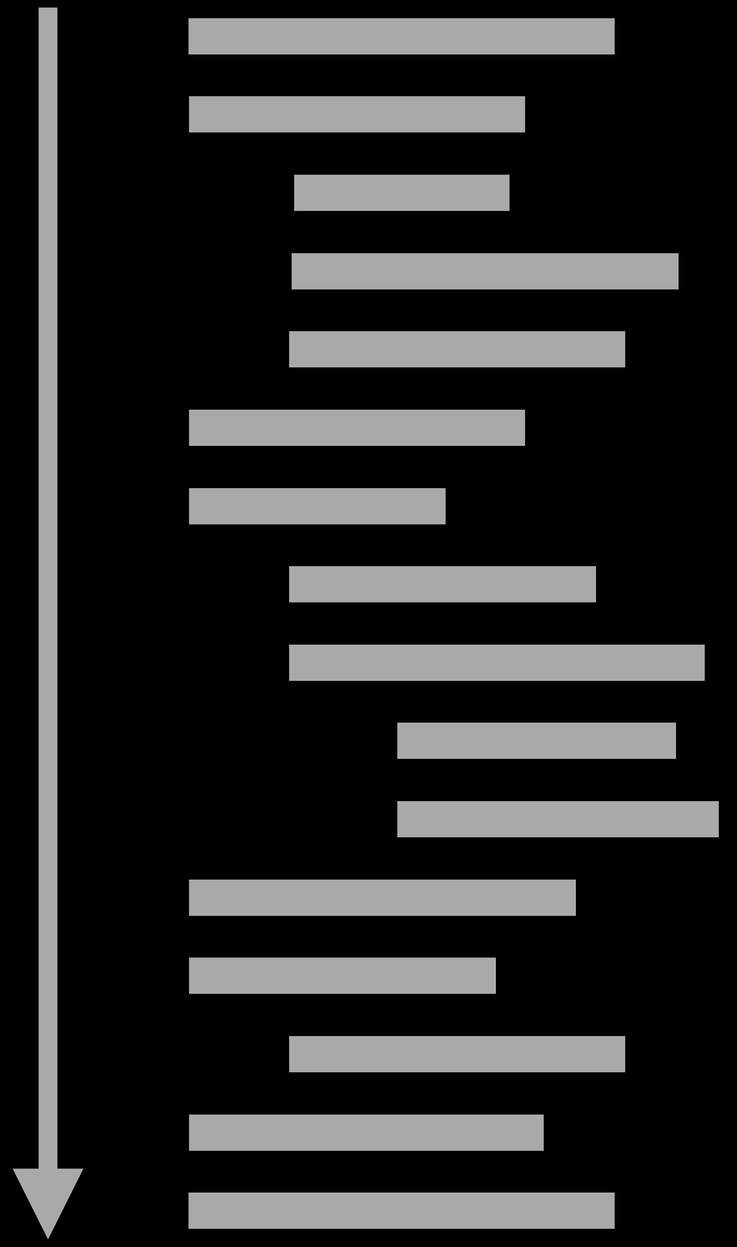
✓ Simple

✓ Retained



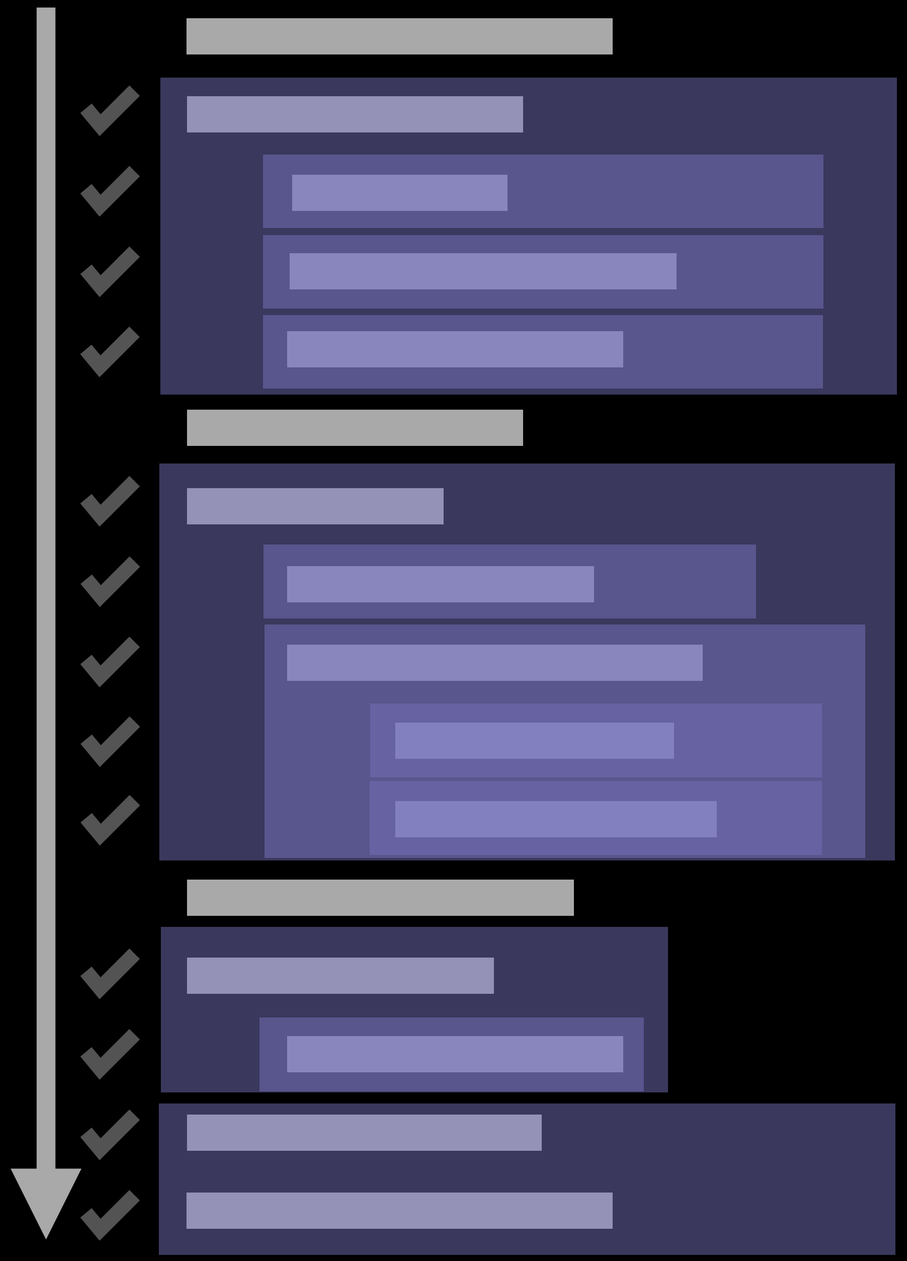
Reactive Mode

- Write all the code to produce 1 frame
- Memoize the slow parts **recursively**



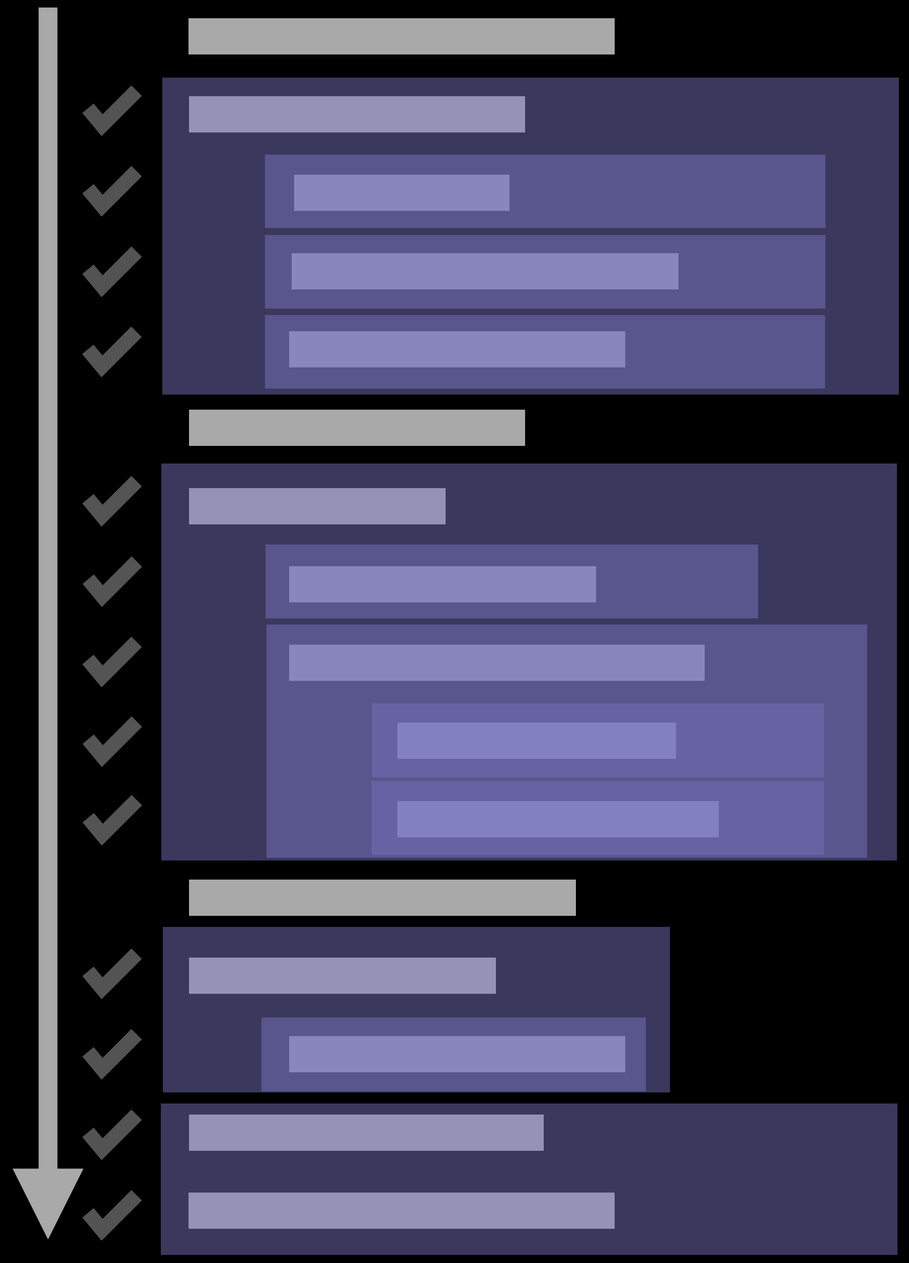
Reactive Mode

- Write all the code to produce 1 frame
- Memoize the slow parts **recursively**



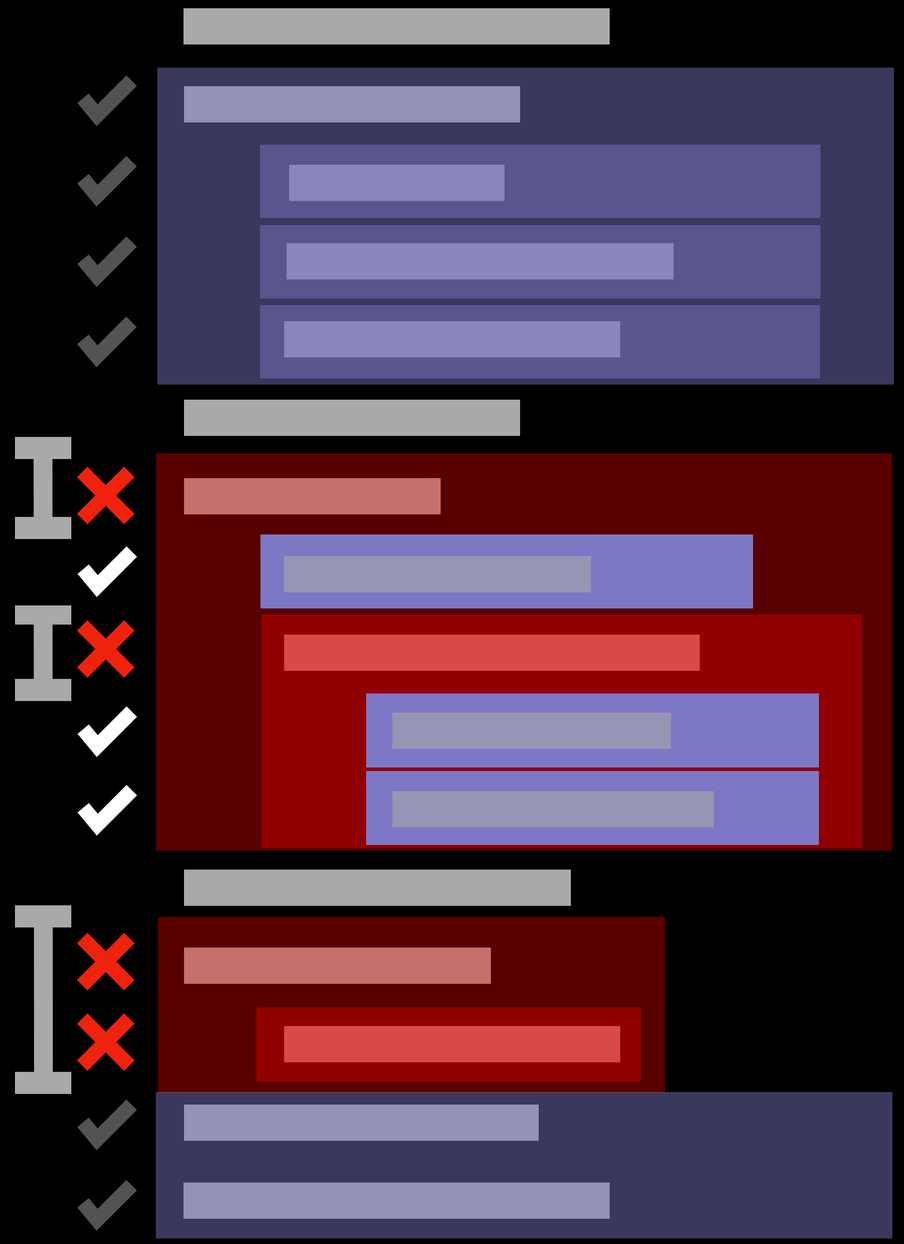
Reactive Mode

- Write all the code to produce 1 frame
- Memoize the slow parts **recursively**
- Re-run subtrees **selectively**



Reactive Mode

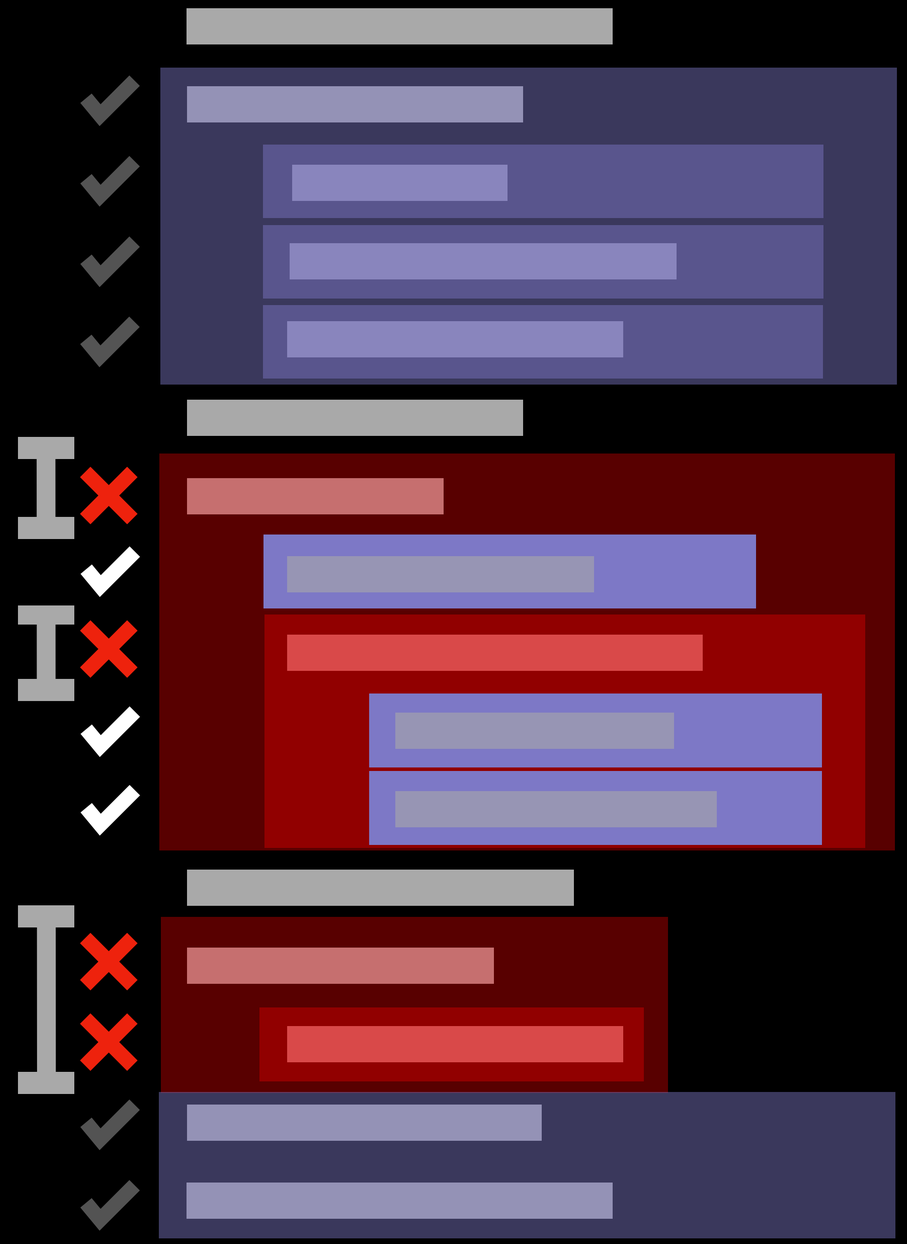
- Write all the code to produce 1 frame
- Memoize the slow parts **recursively**
- Re-run subtrees **selectively**



Reactive Mode

- Write all the code to produce 1 frame
- Memoize the slow parts **recursively**
- Re-run subtrees **selectively**

⚠ Complex

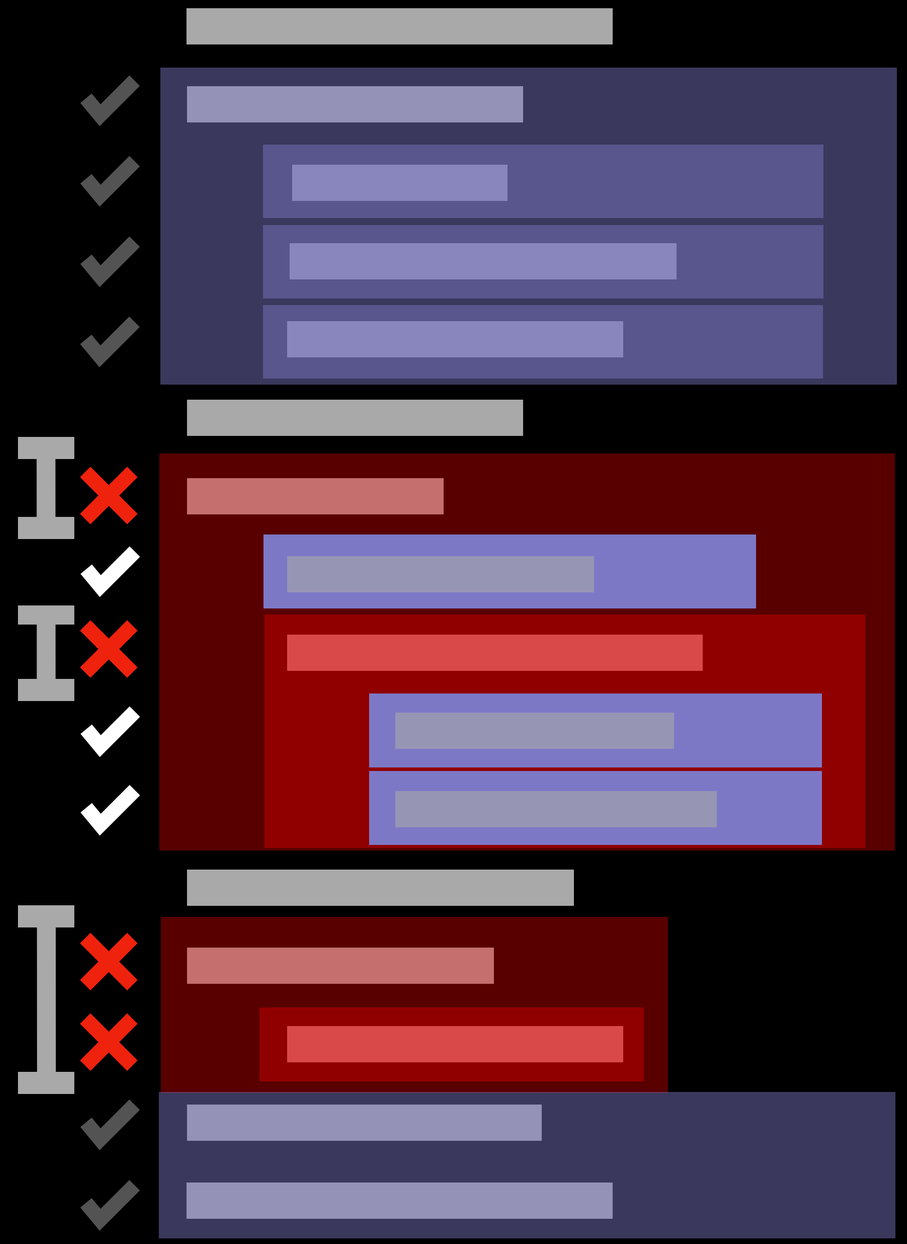


Reactive Mode

- Write all the code to produce 1 frame
- Memoize the slow parts **recursively**
- Re-run subtrees **selectively**

⚠ Complex

✓ Very Retained?

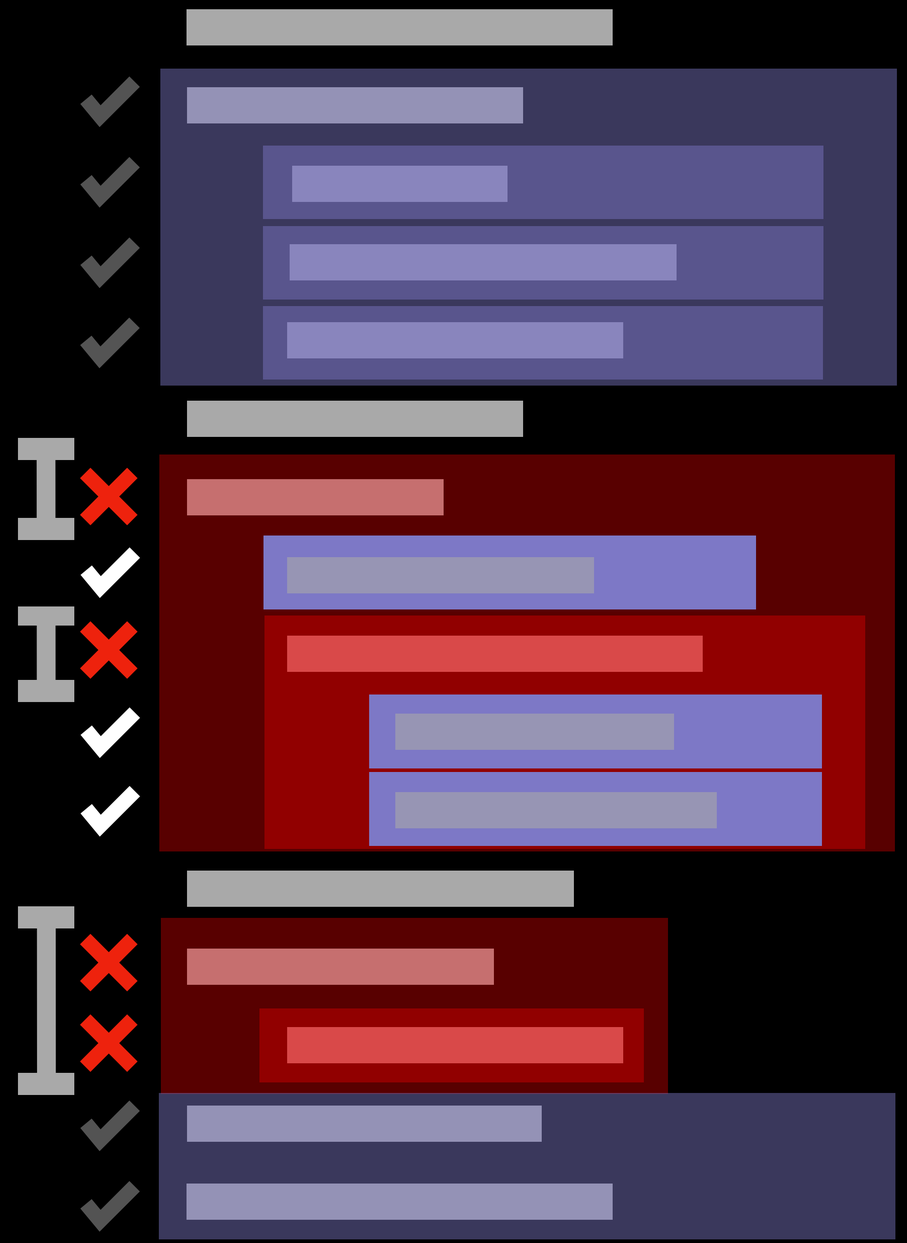


Reactive Mode

- Write all the code to produce 1 frame
- Memoize the slow parts **recursively**
- Re-run subtrees **selectively**

⚠ Complex

⚠ Turing complete



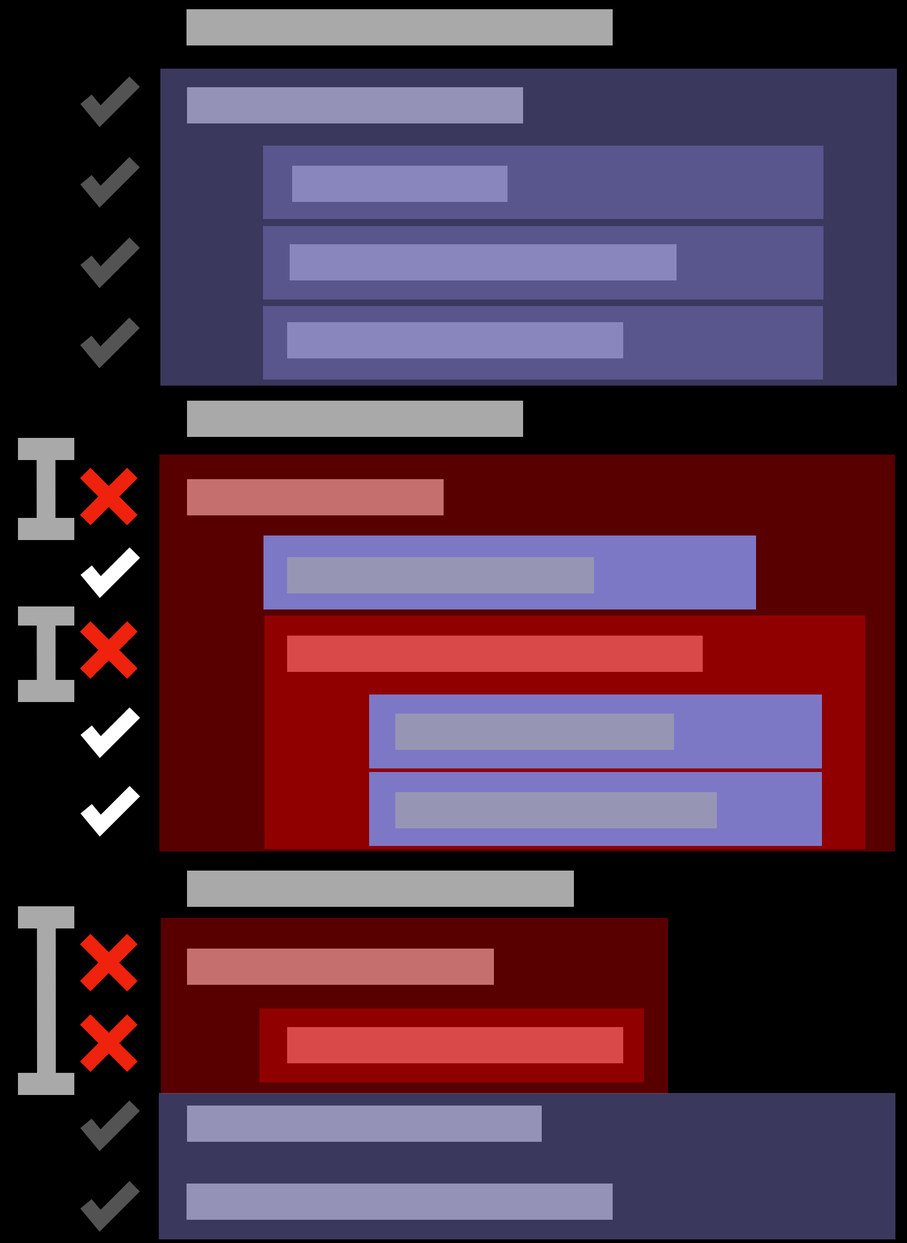
Reactive Mode

- Write all the code to produce 1 frame
- Memoize the slow parts **recursively**
- Re-run subtrees **selectively**

⚠ Complex

⚠ Turing complete

⚠ Immutable data



Web / React

Web / React

Build **reactive mode trees**

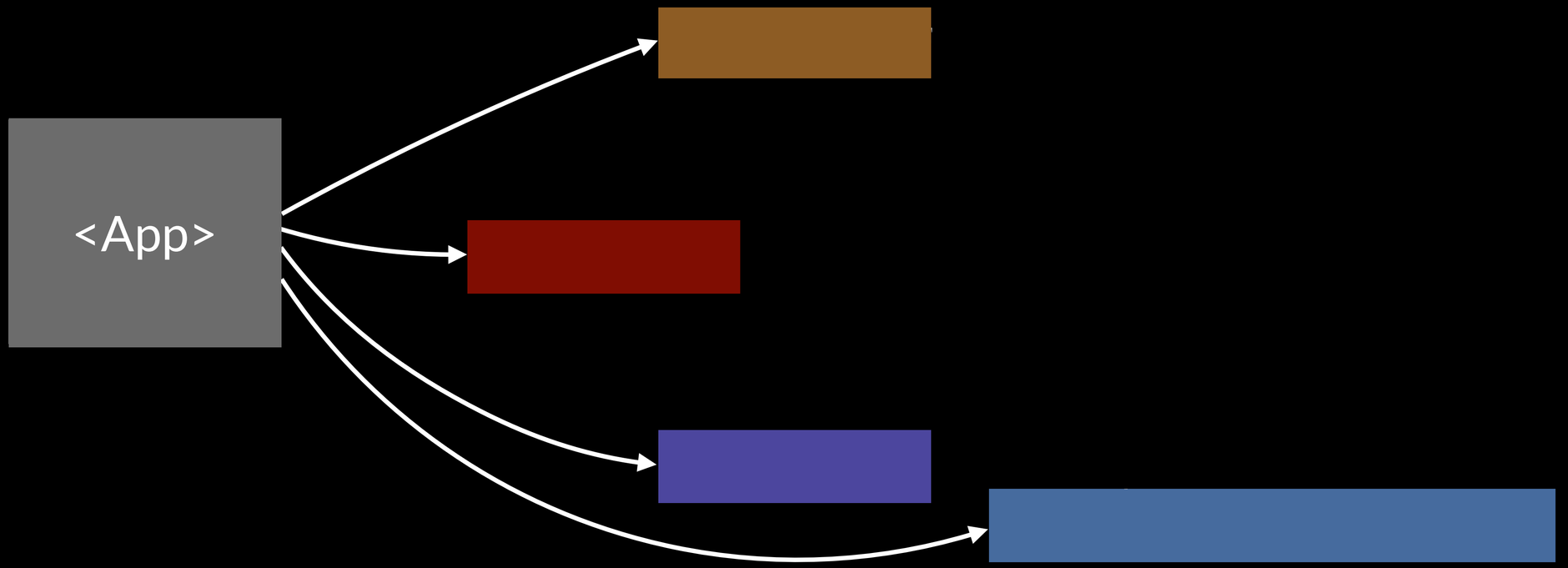
Web / React

Build **reactive mode trees**

<App>

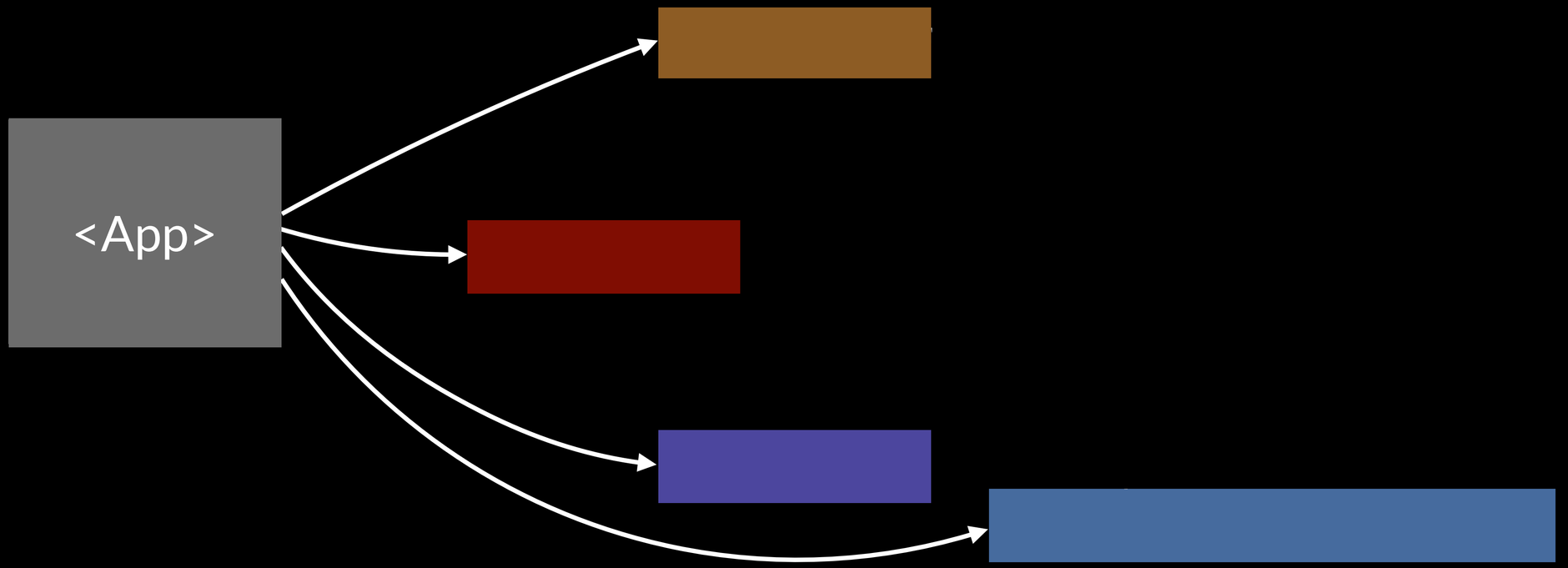
Web / React

Build **reactive mode trees**



Web / React

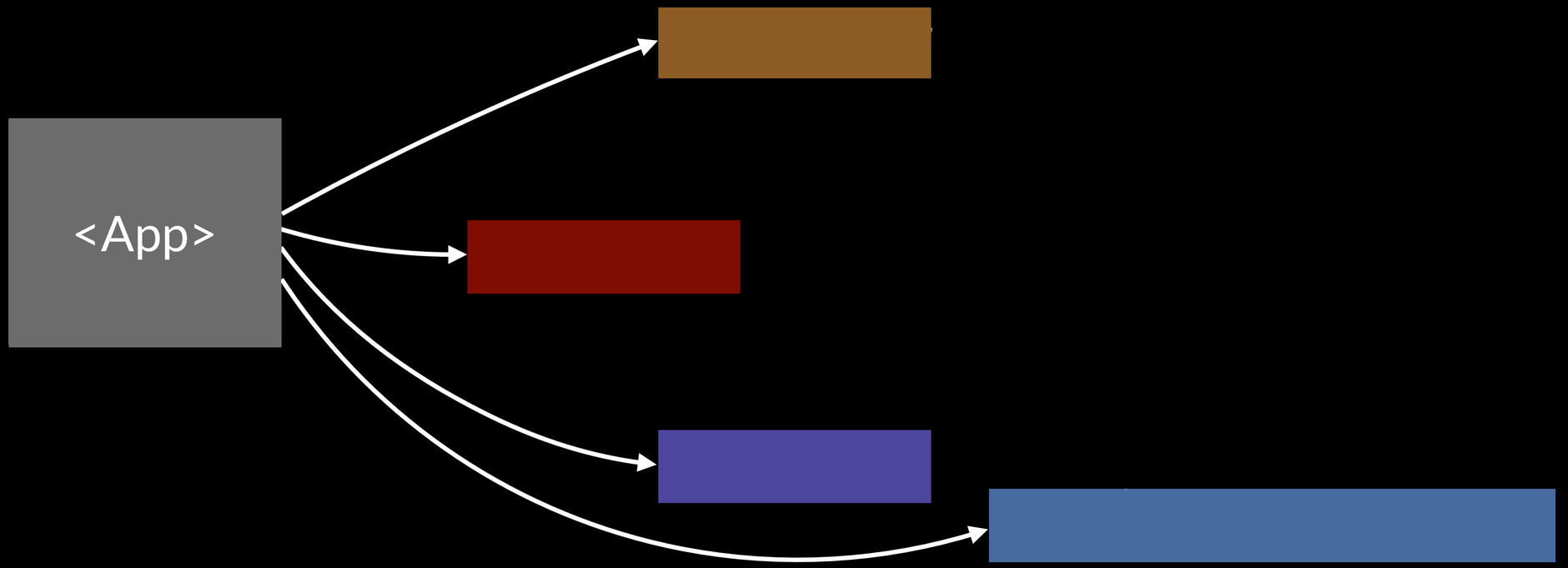
Build **reactive mode trees**



Data flows from a source into **components**

Web / React

Build **reactive mode trees**

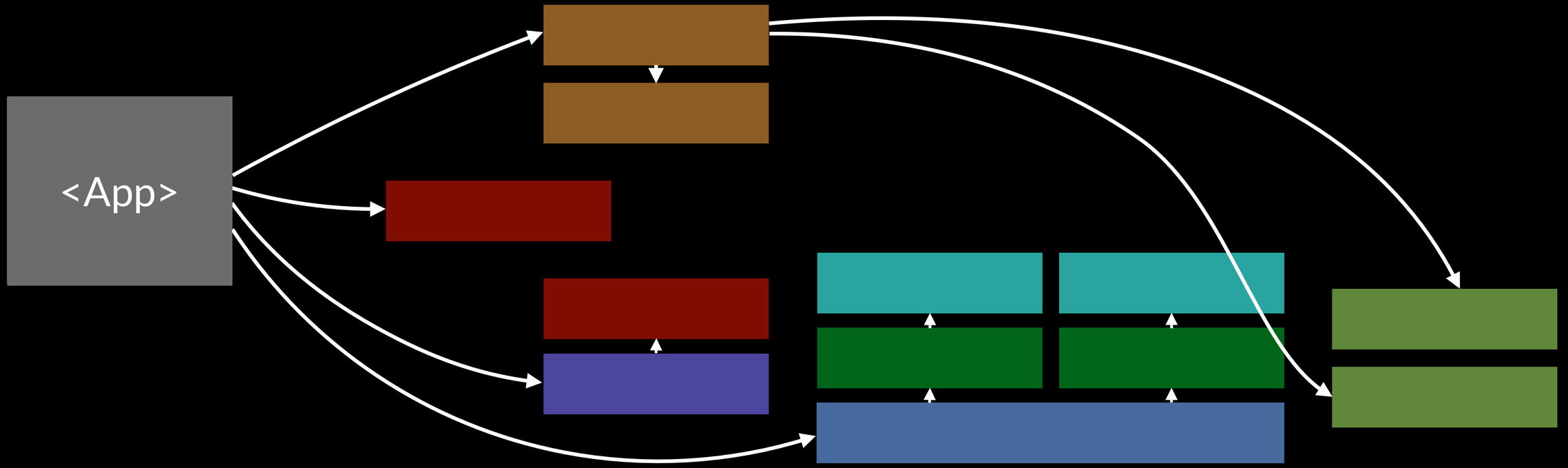


Data flows from a source into **components**

= **Functions** return child components at **run-time**

Web / React

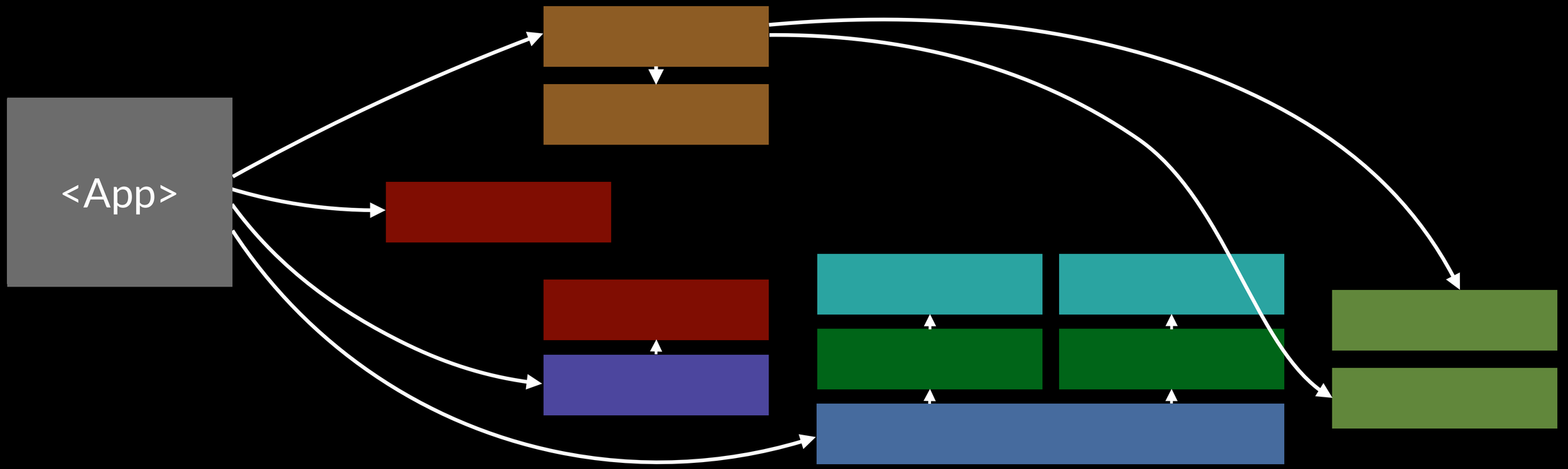
Build **reactive mode trees**



Data flows from a source into **components**

= **Functions** return child components at **run-time**

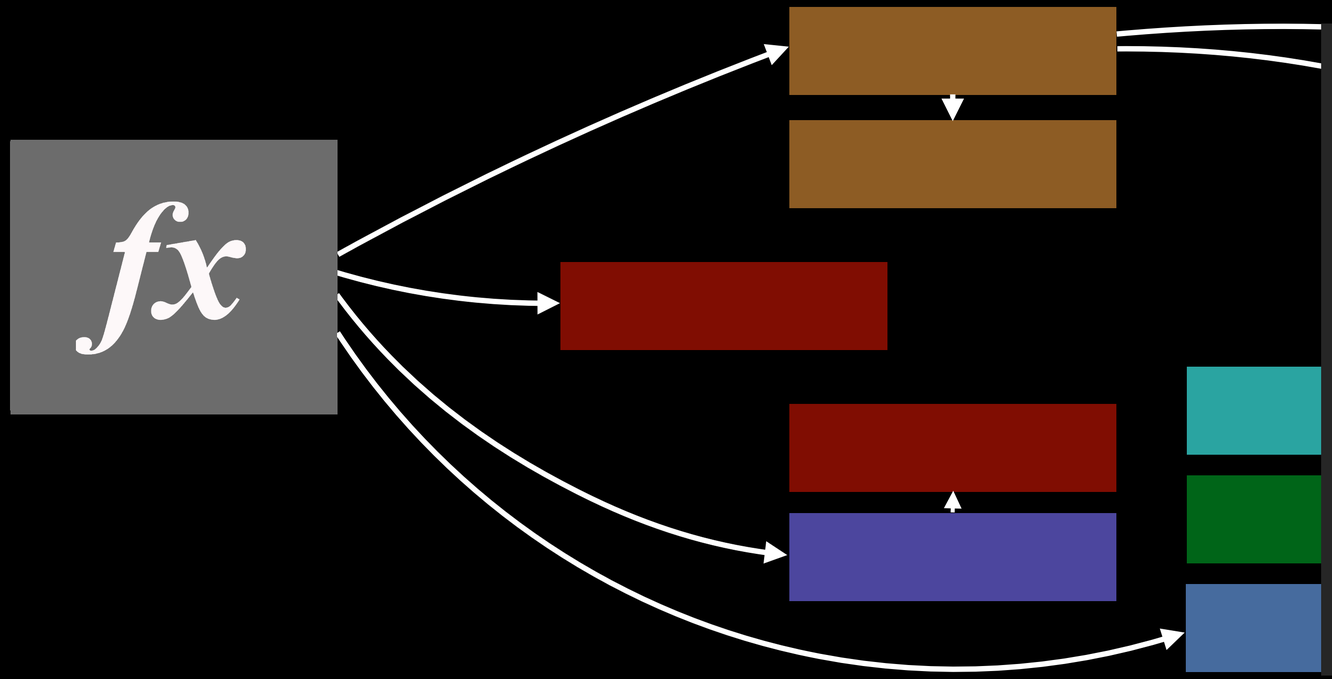
Web / React



Data flows from a source into **components**

= **Functions** declare child components at **run-time**

Web / React



JSX

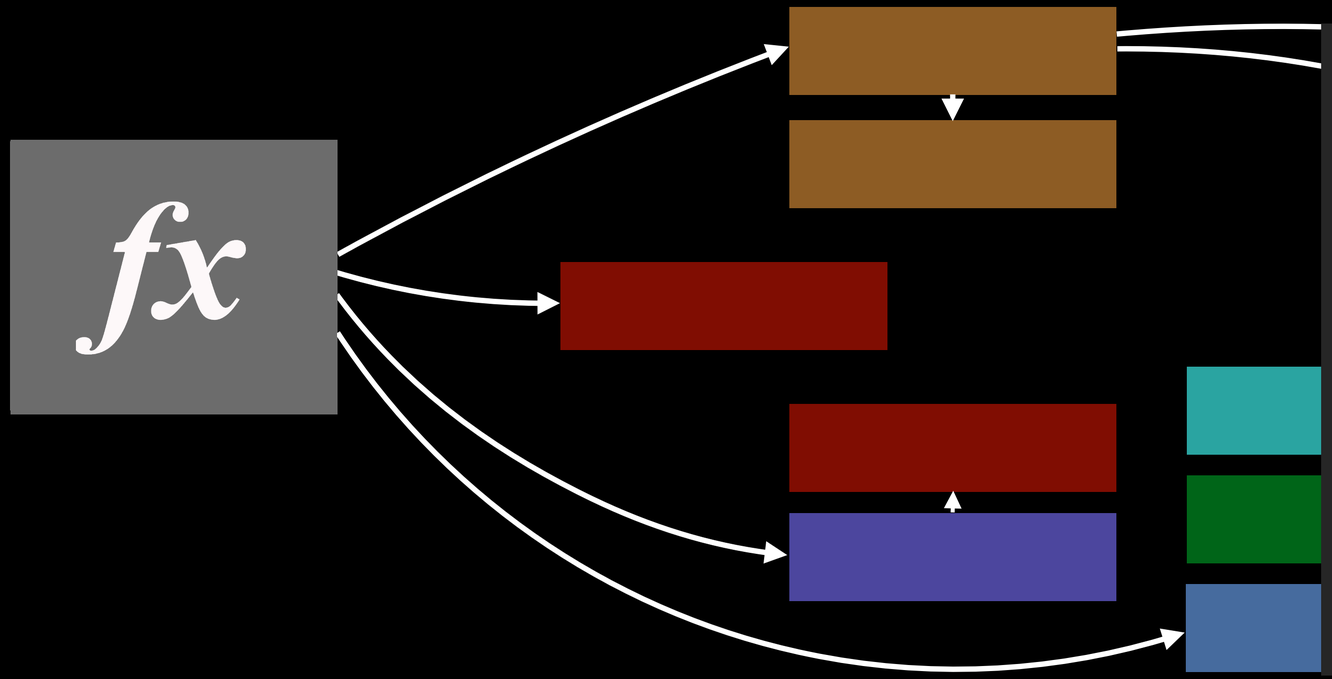
```
return (<>
  <Red />
  <Orange>
    <Green />
    <Green foo={…} />
  </Orange>
  <Purple bar={…} />
  <Blue />
</>)
```

Data flows from a source into **components**

= **Functions** declare child components at **run-time**

Web / React

Tree **knits itself together**
at runtime



JSX

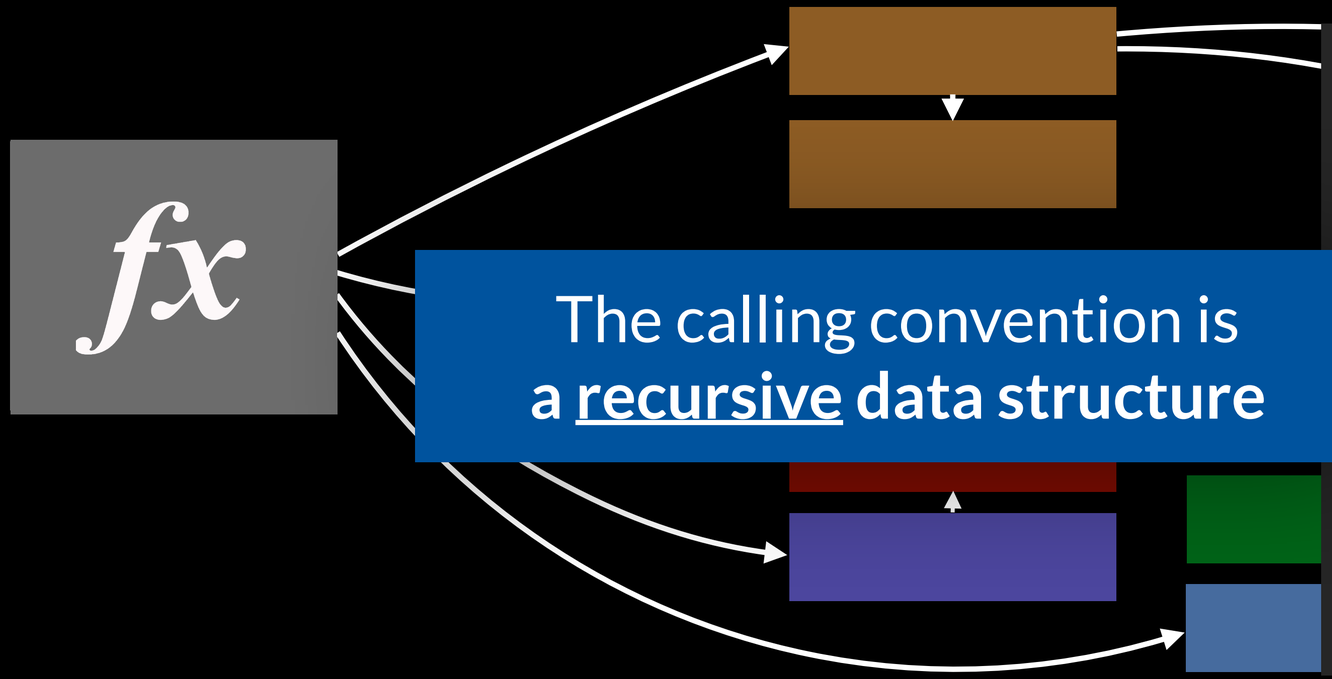
```
return (<>
  <Red />
  <Orange>
    <Green />
    <Green foo={…} />
  </Orange>
  <Purple bar={…} />
  <Blue />
</>)
```

Data flows from a source into **components**

= **Functions** declare child components at **run-time**

Web / React

Tree **knits itself together**
at runtime



The calling convention is
a recursive data structure

JSX

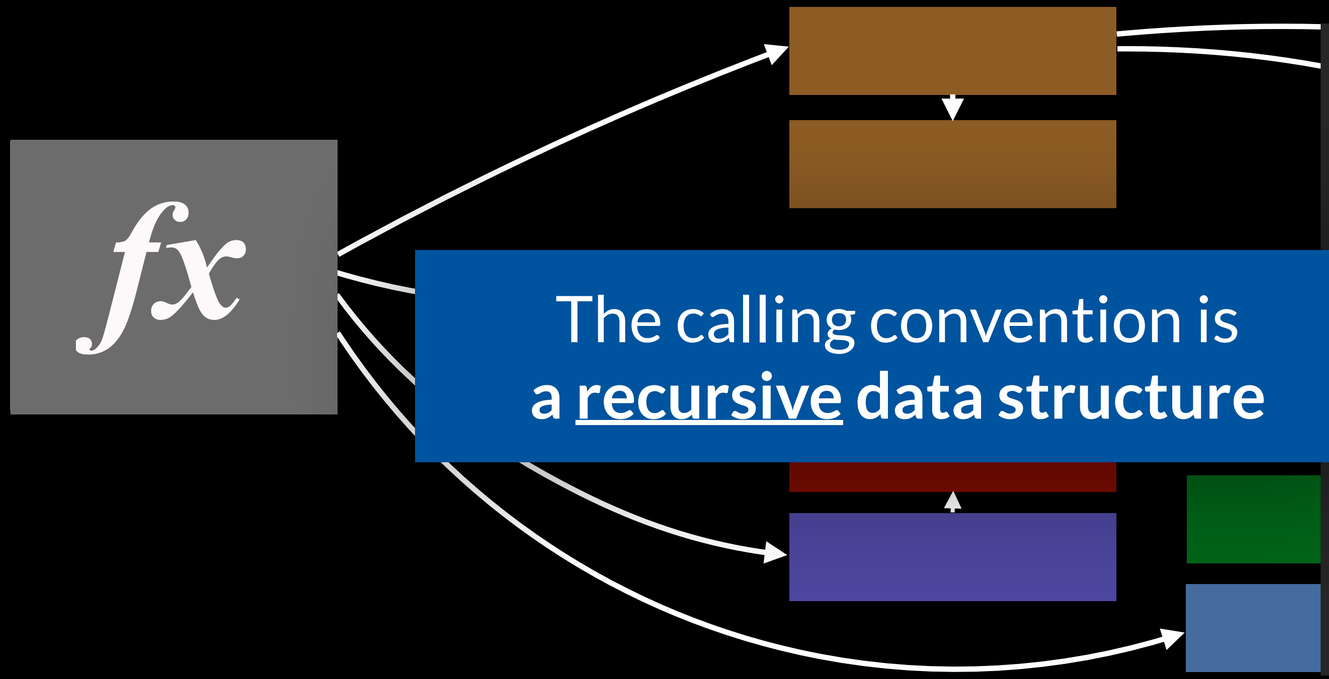
```
return (<>
  <Red />
  <Orange>
    <Green />
    <Green foo={…} />
  </Orange>
  <Purple bar={…} />
  <Blue />
</>)
```

Data flows from a source into **components**

= **Functions** declare child components at **run-time**

Web / React

Tree **knits itself together**
at runtime



The calling convention is
a recursive data structure

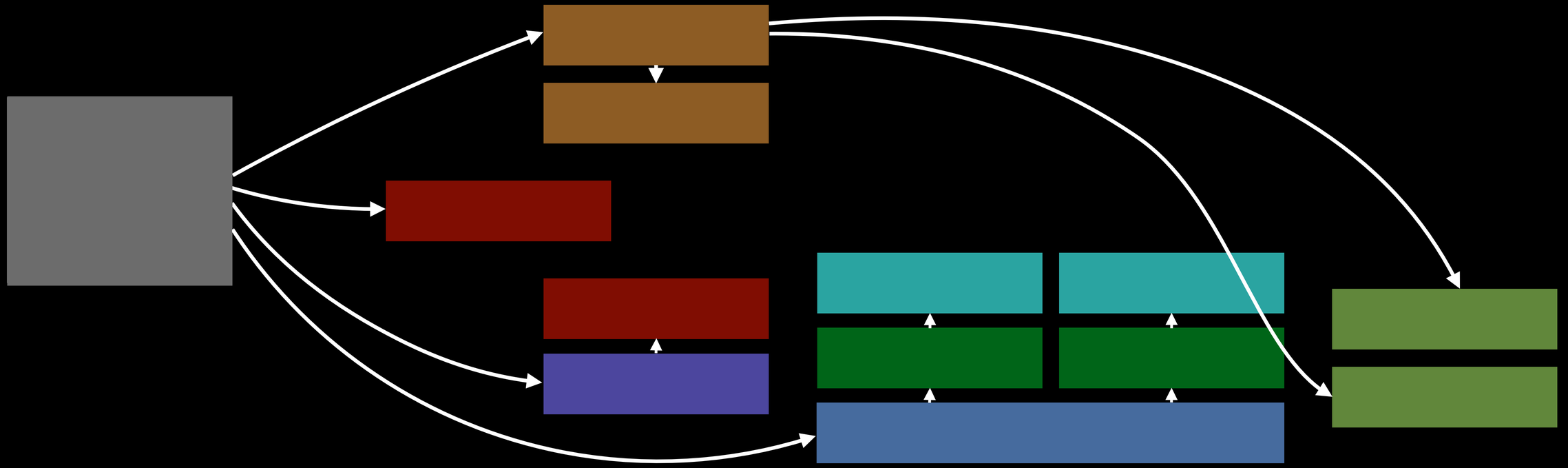
```
return [                                  Read as
  {fn: Red},
  {fn: Orange, args: [{…
    {fn: Green},
    {fn: Green, args: […]},
  …}]},
  {fn: Purple, args: […]},
  {fn: Blue},
]
```

Data flows from a source into **components**

= **Functions** declare child components at **run-time**

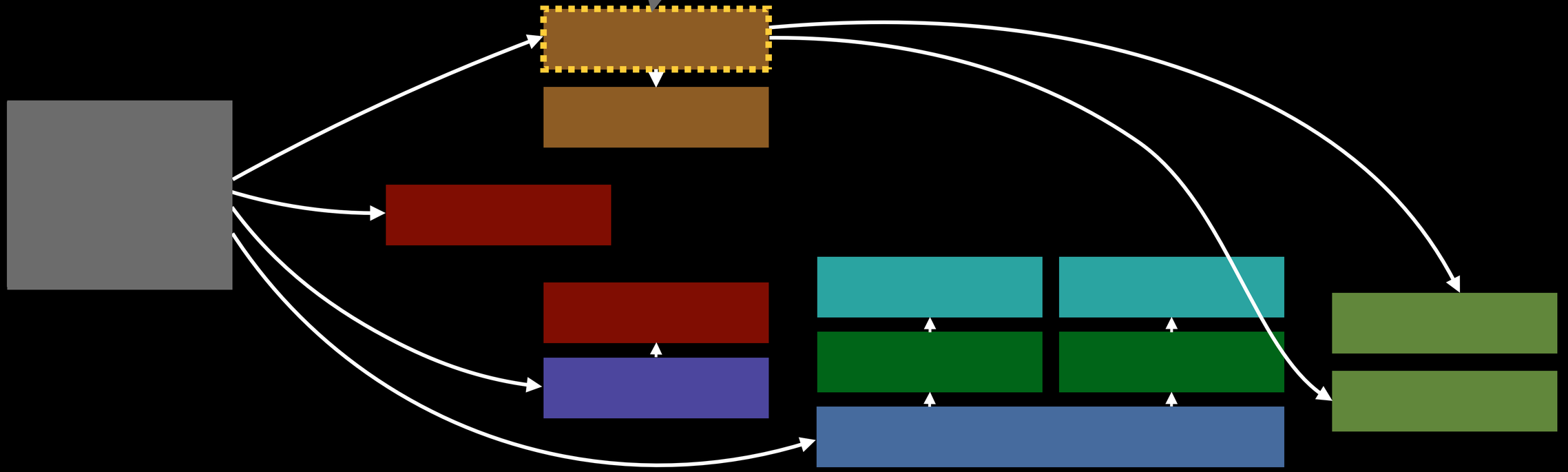
Web / React

Tree **knits itself together**
at runtime



Web / React

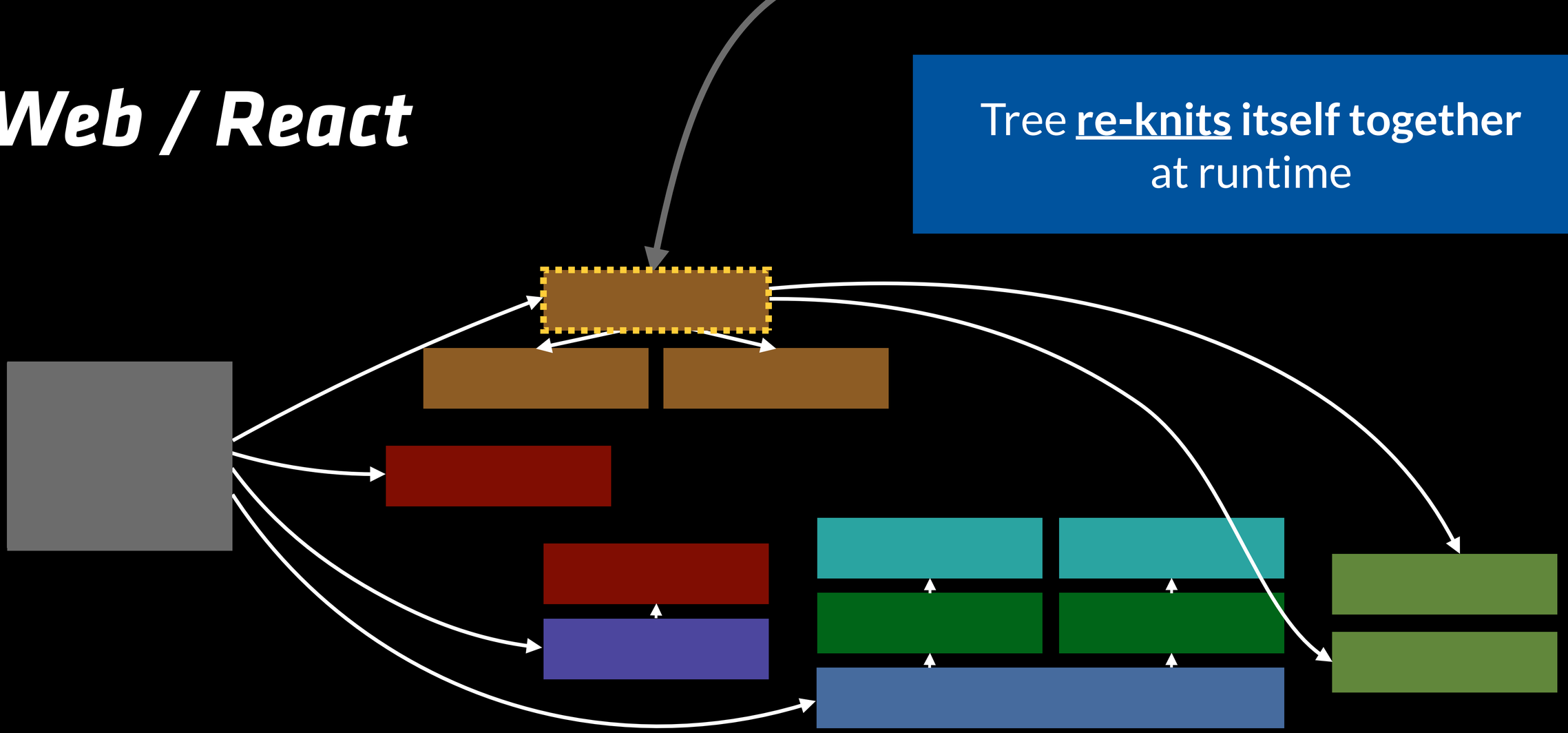
Tree **knits itself together**
at runtime



Components whose **state changed** are **re-run**

Web / React

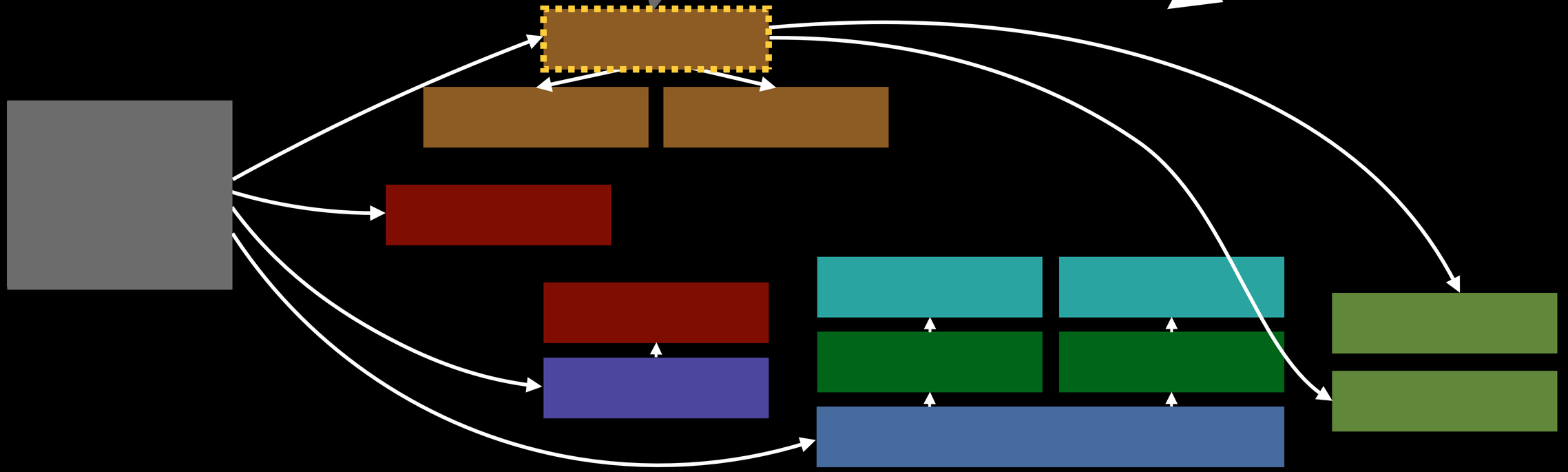
Tree **re-knits itself together** at runtime



Components whose **state changed** are **re-run**

Web / React

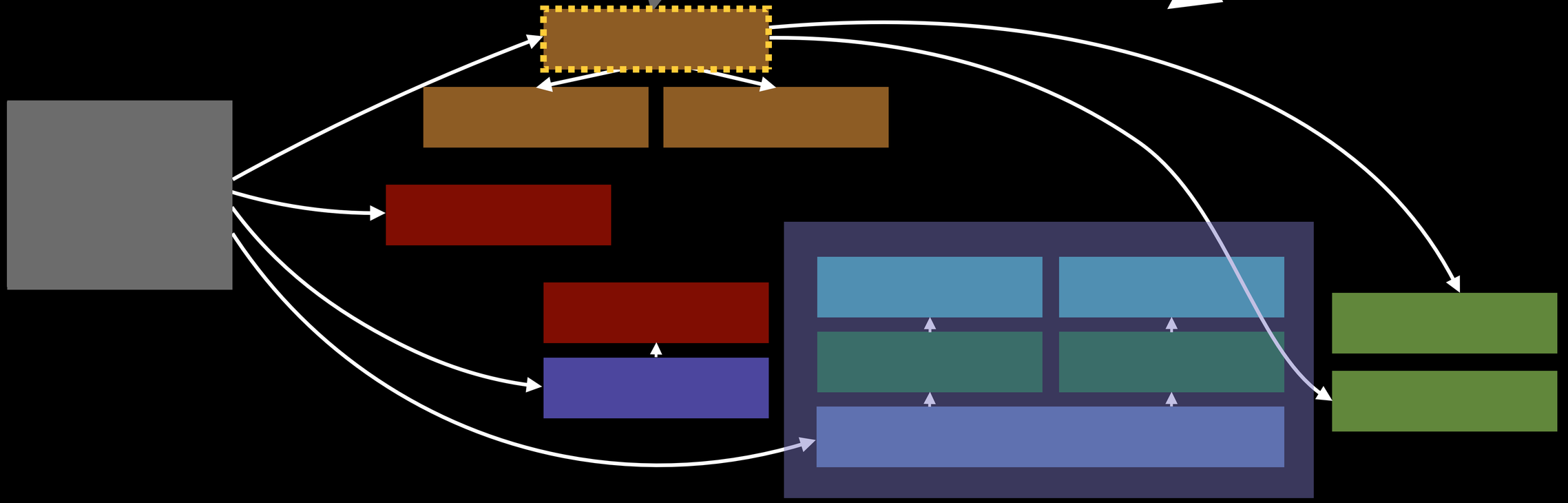
The **deeper in the tree**
the **more frequently** it changes



Components whose **state changed** are **re-run**

Web / React

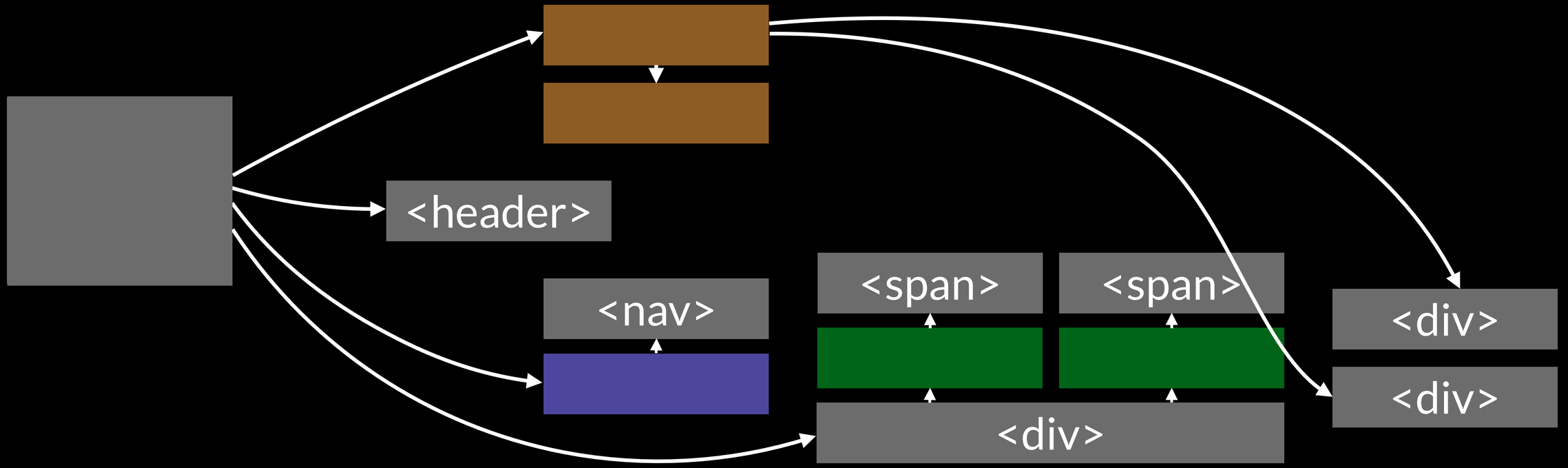
The **deeper in the tree**
the **more frequently** it changes



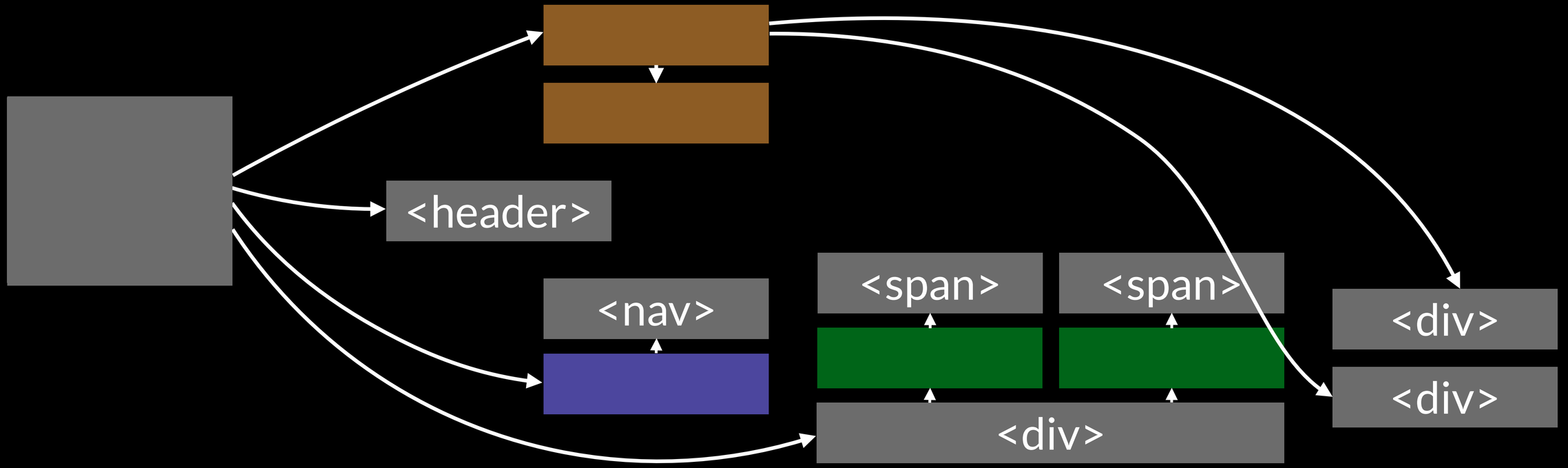
Components whose **state changed** are **re-run**

Memoize components to avoid updating slow sub-trees

Web / React

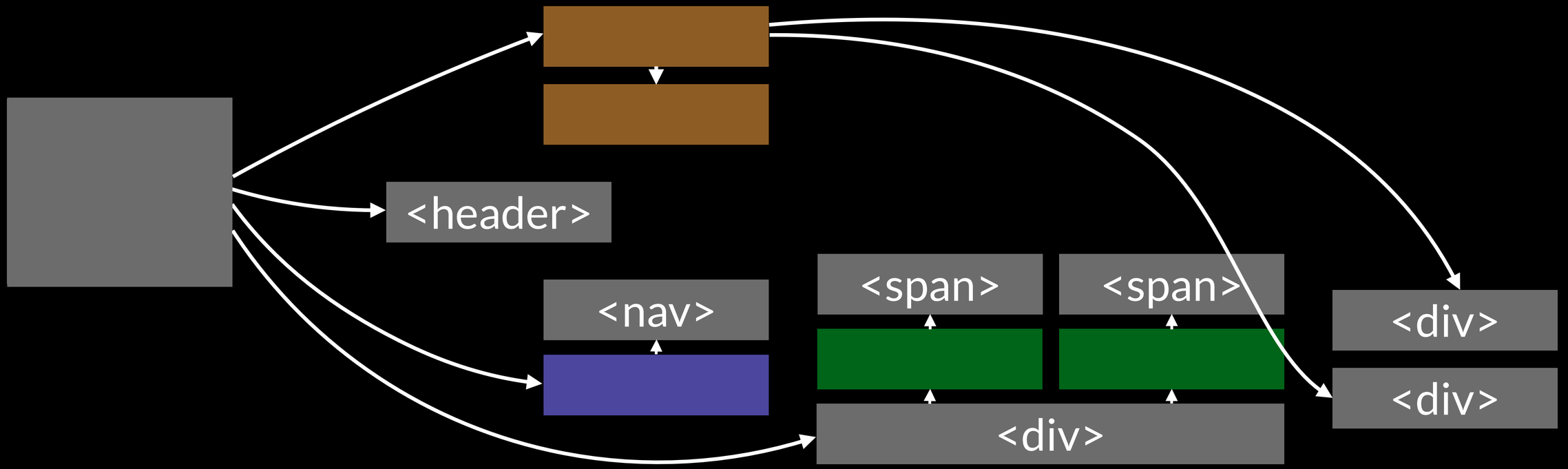


Web / React



React **proxies** to native HTML

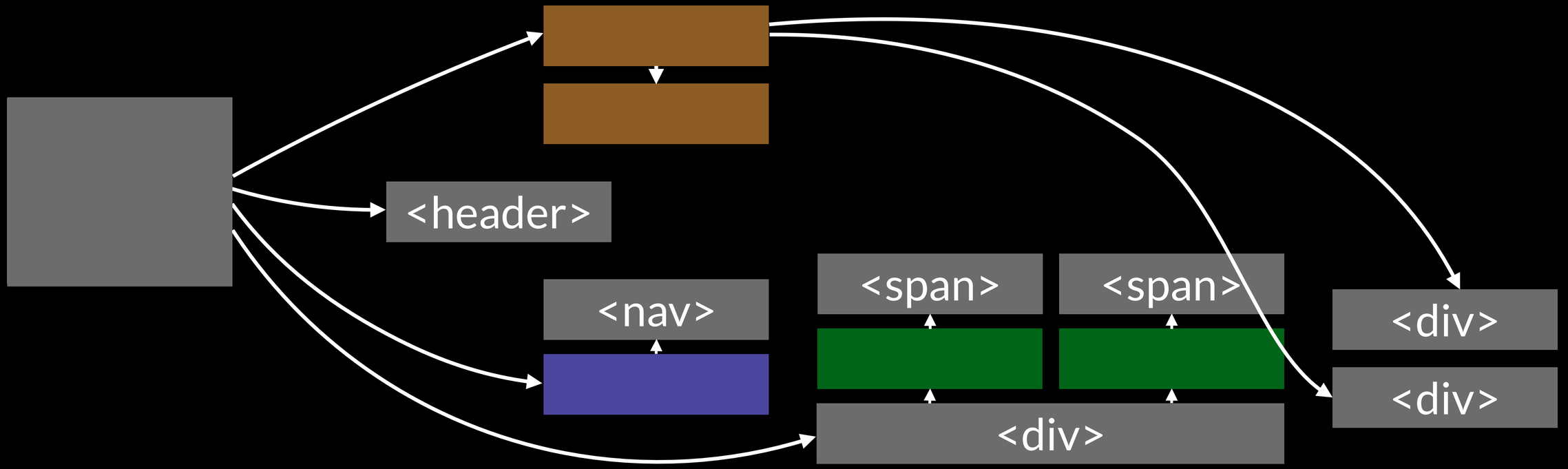
Web / React



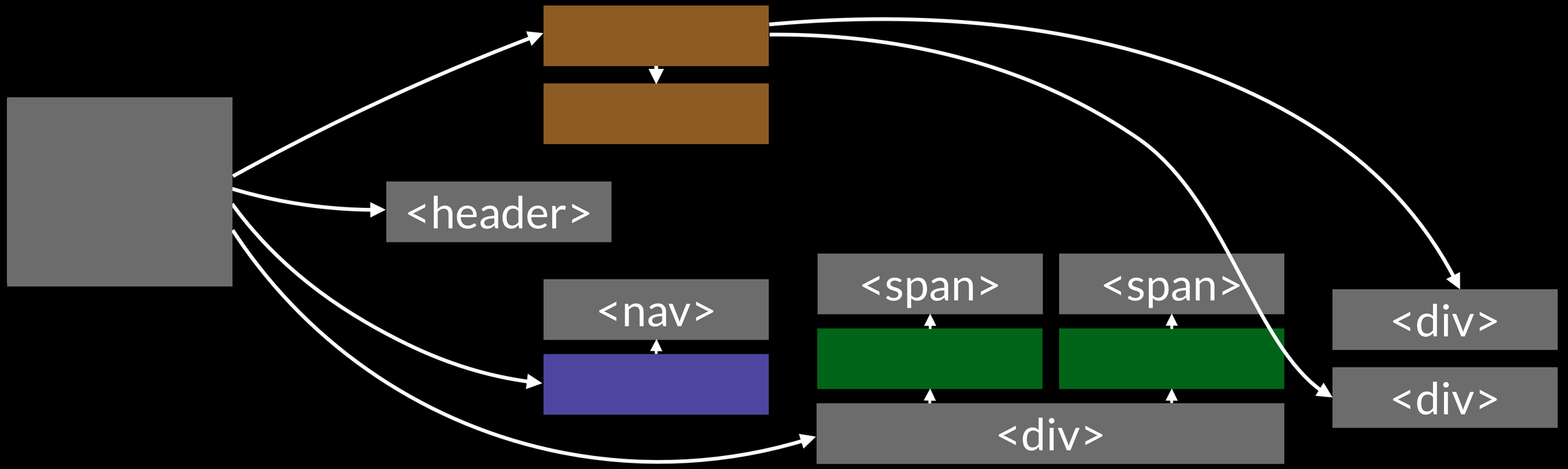
React **proxies** to native HTML

HTML <elements>
are **special components**

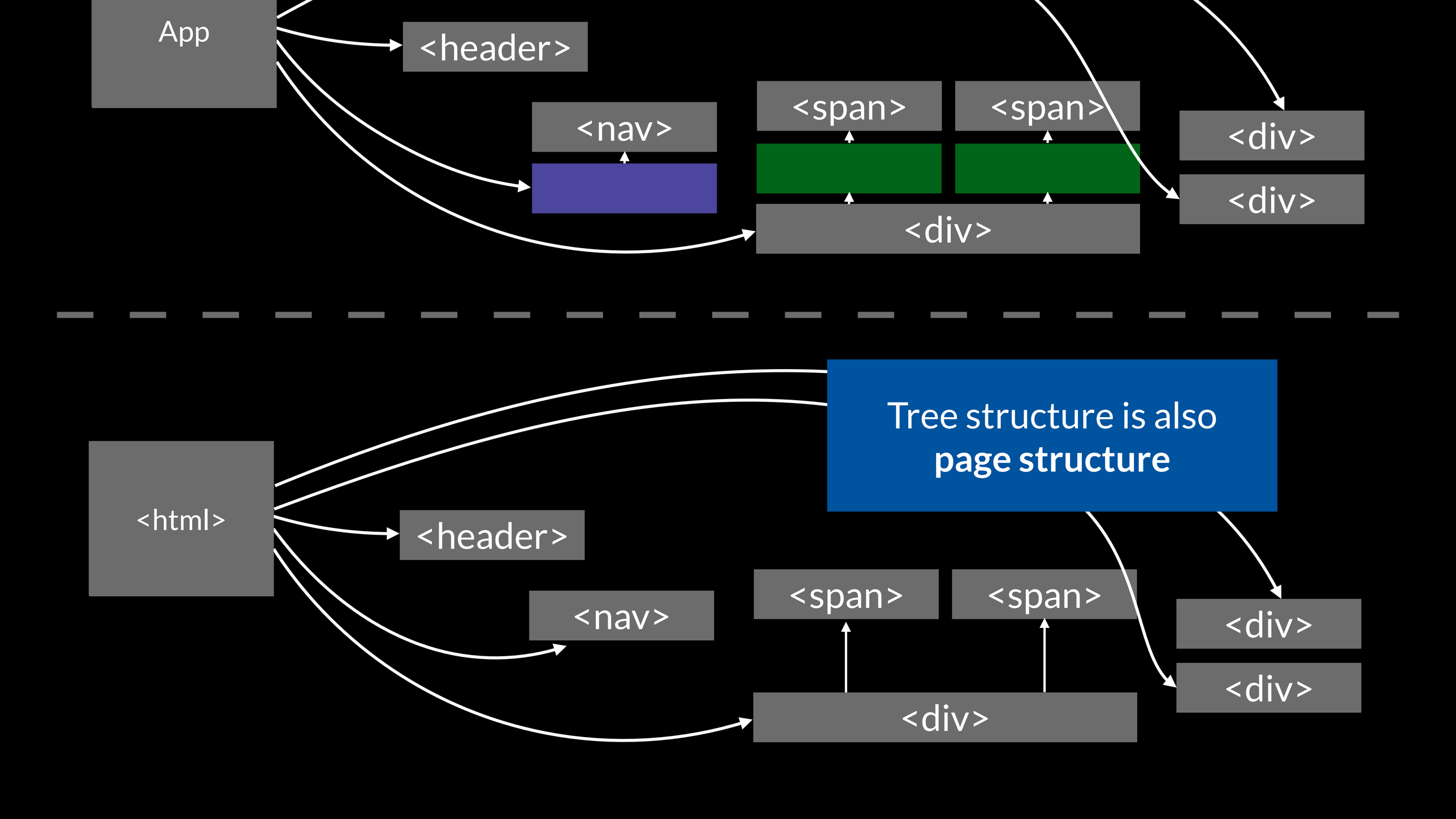
Web / React



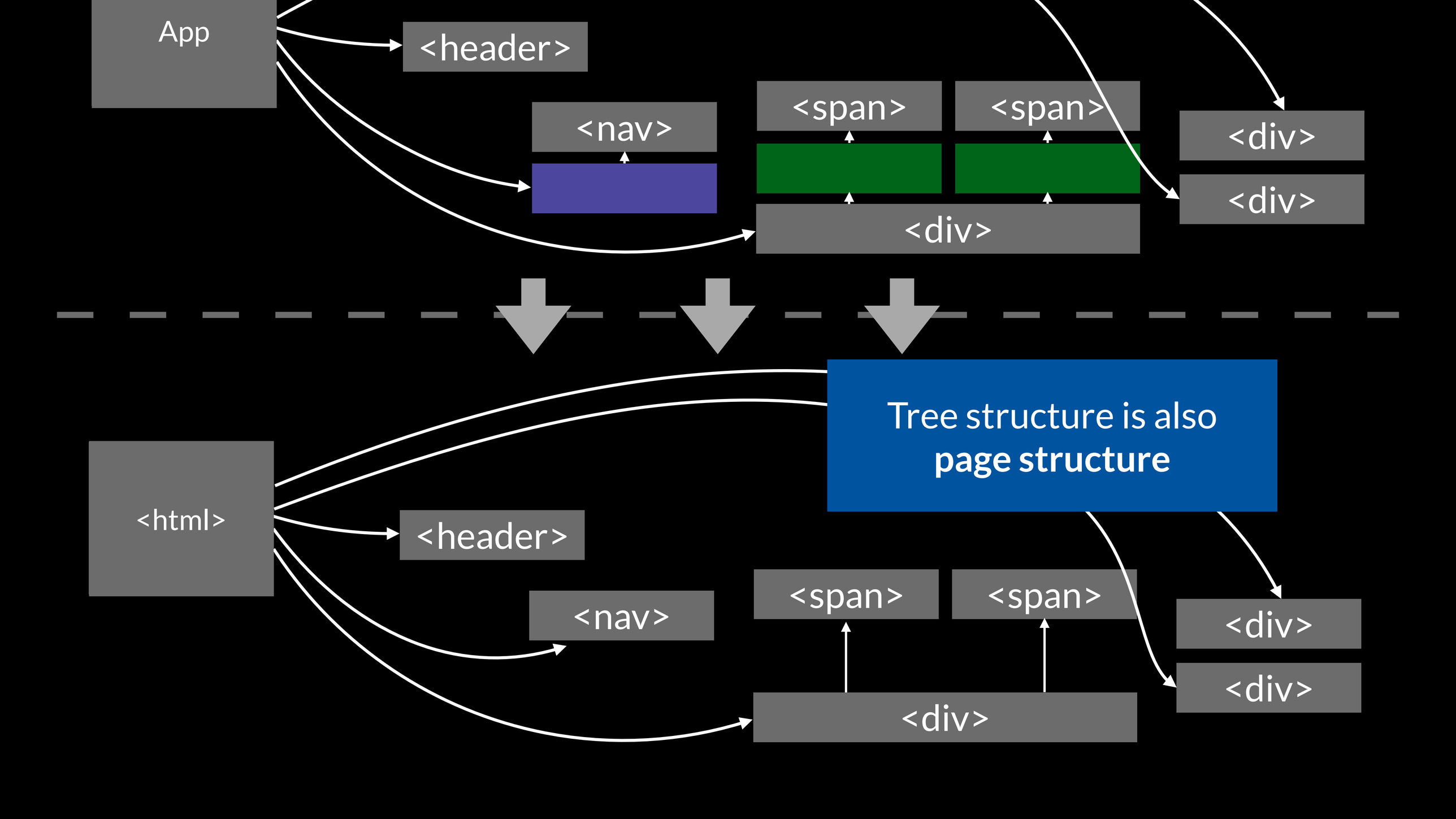
Web / React



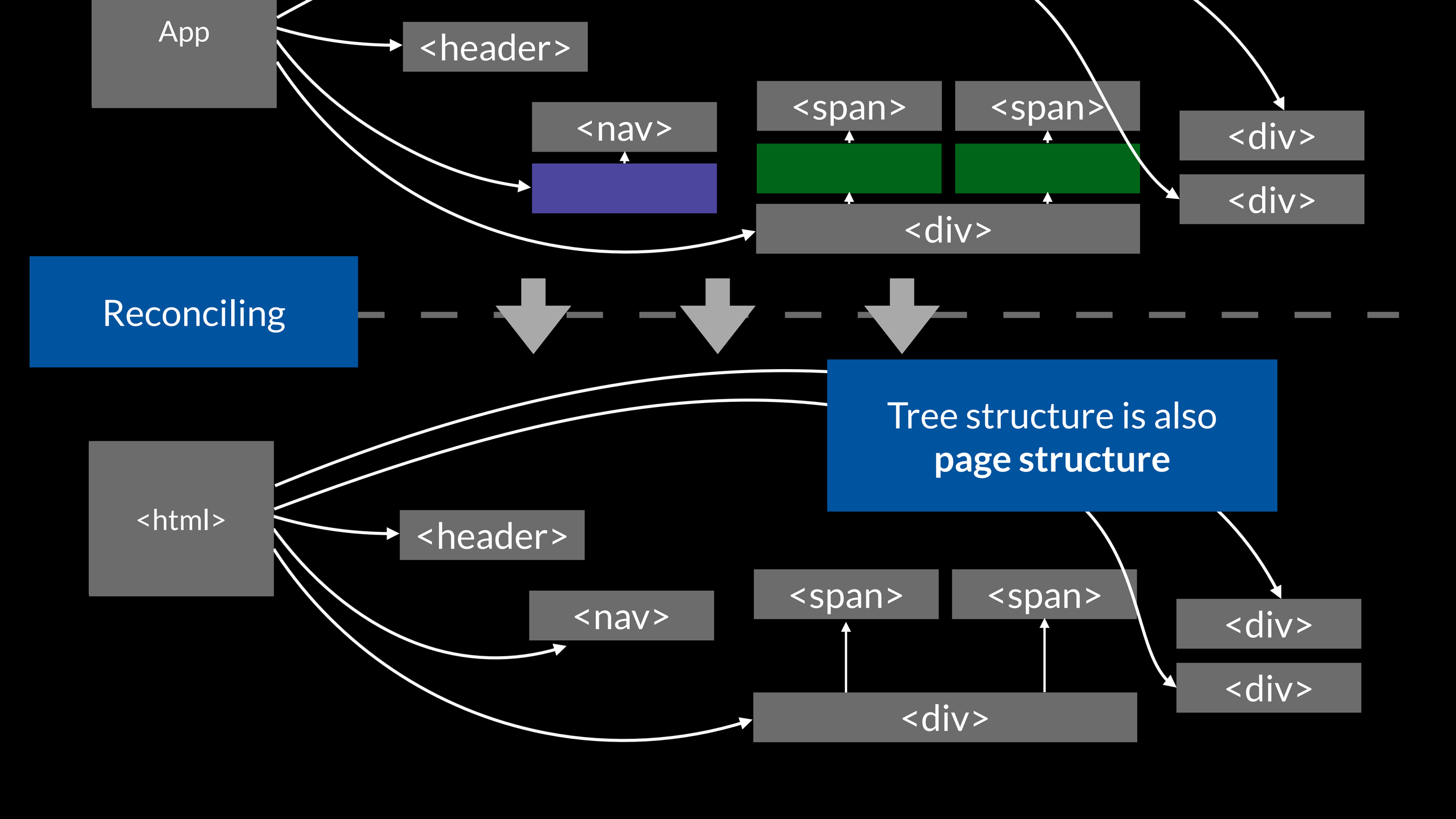
Tree structure is also
page structure


App
<header>
<nav>

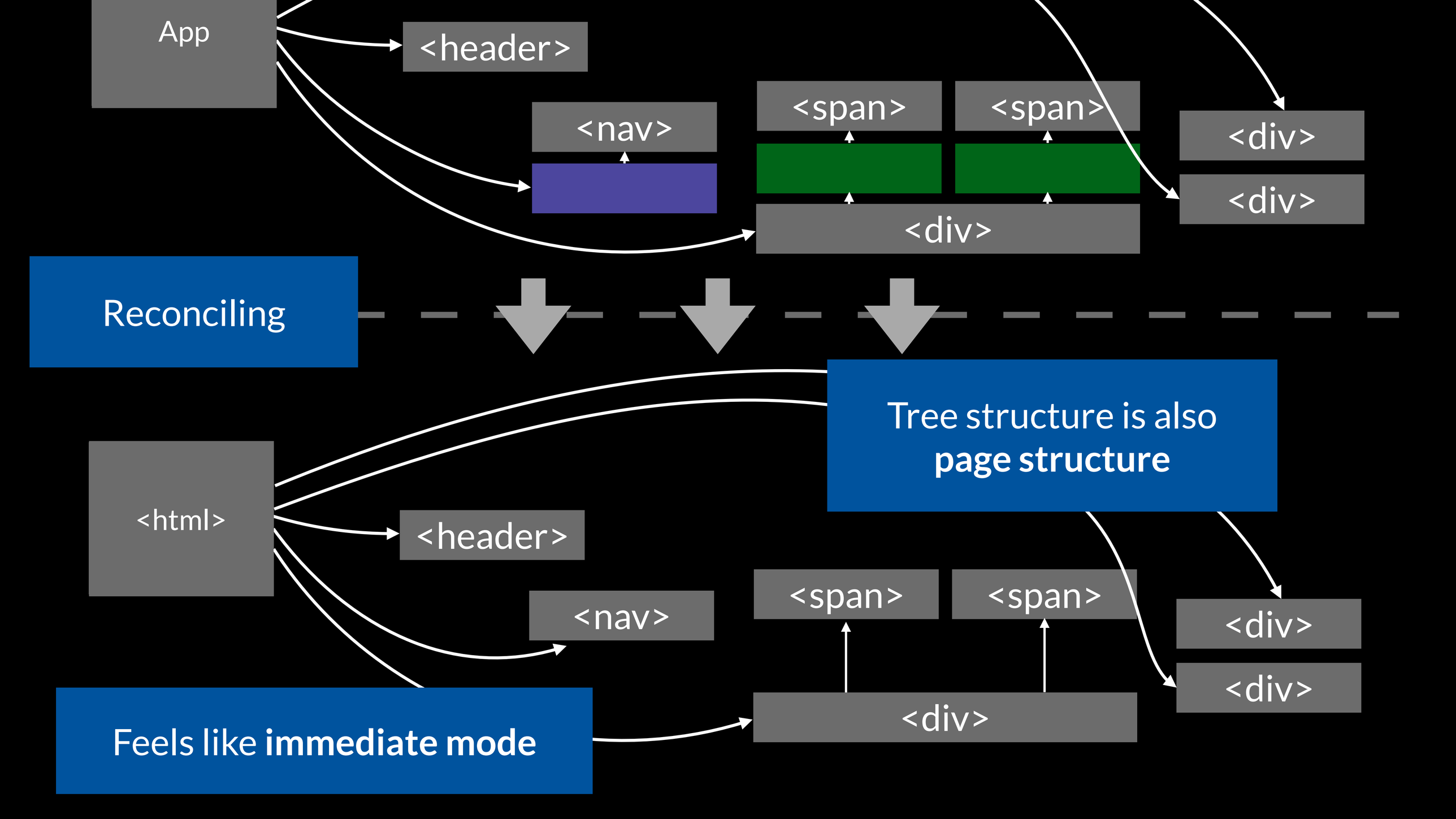
Tree structure is also
page structure
<html>
<header>
<nav>


App
<header>
<nav>

Tree structure is also
page structure
<html>
<header>
<nav>


App
<header>
<nav>

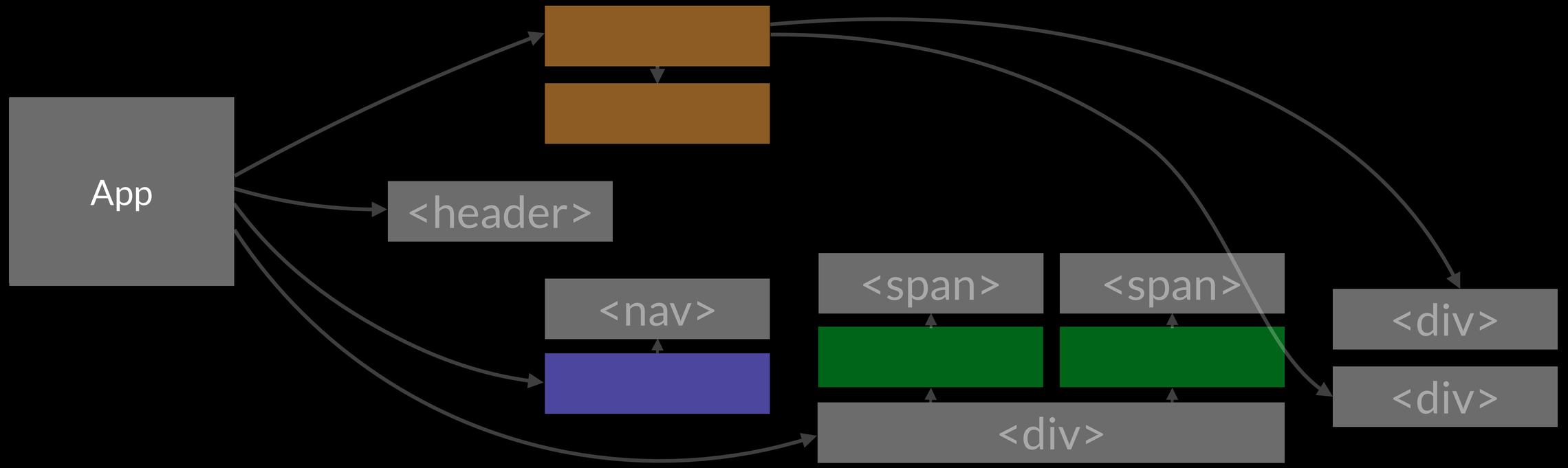
Reconciling
Tree structure is also
page structure
<html>
<header>
<nav>

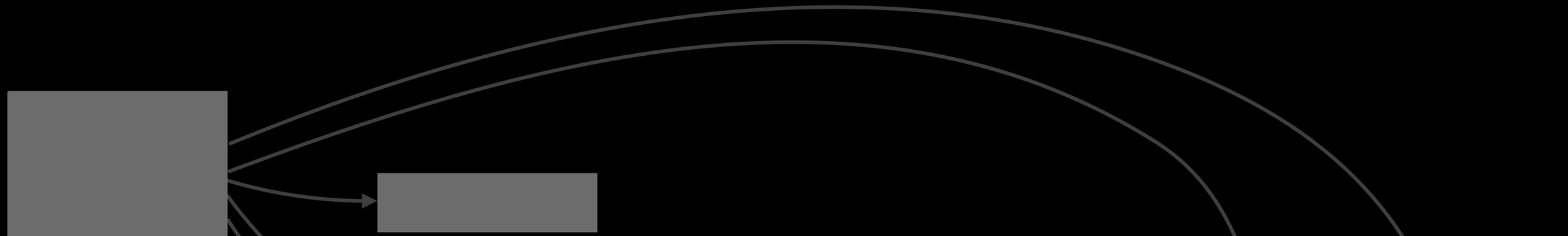

App
<header>
<nav>

Reconciling
Tree structure is also
page structure
<html>
<header>
<nav>

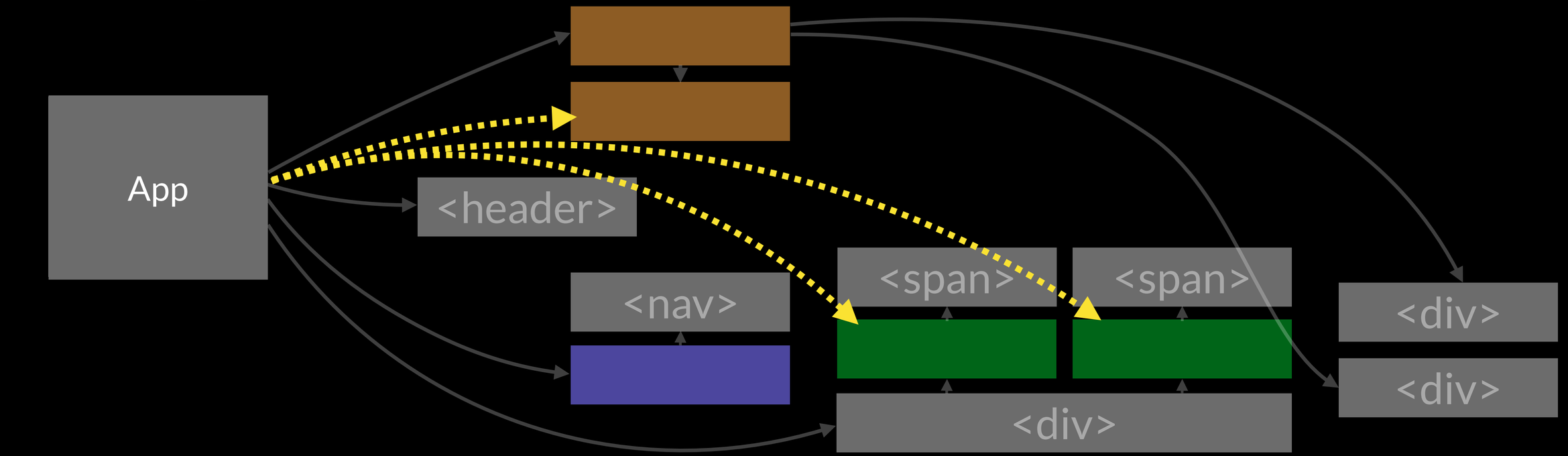
Feels like **immediate mode**

Web / React

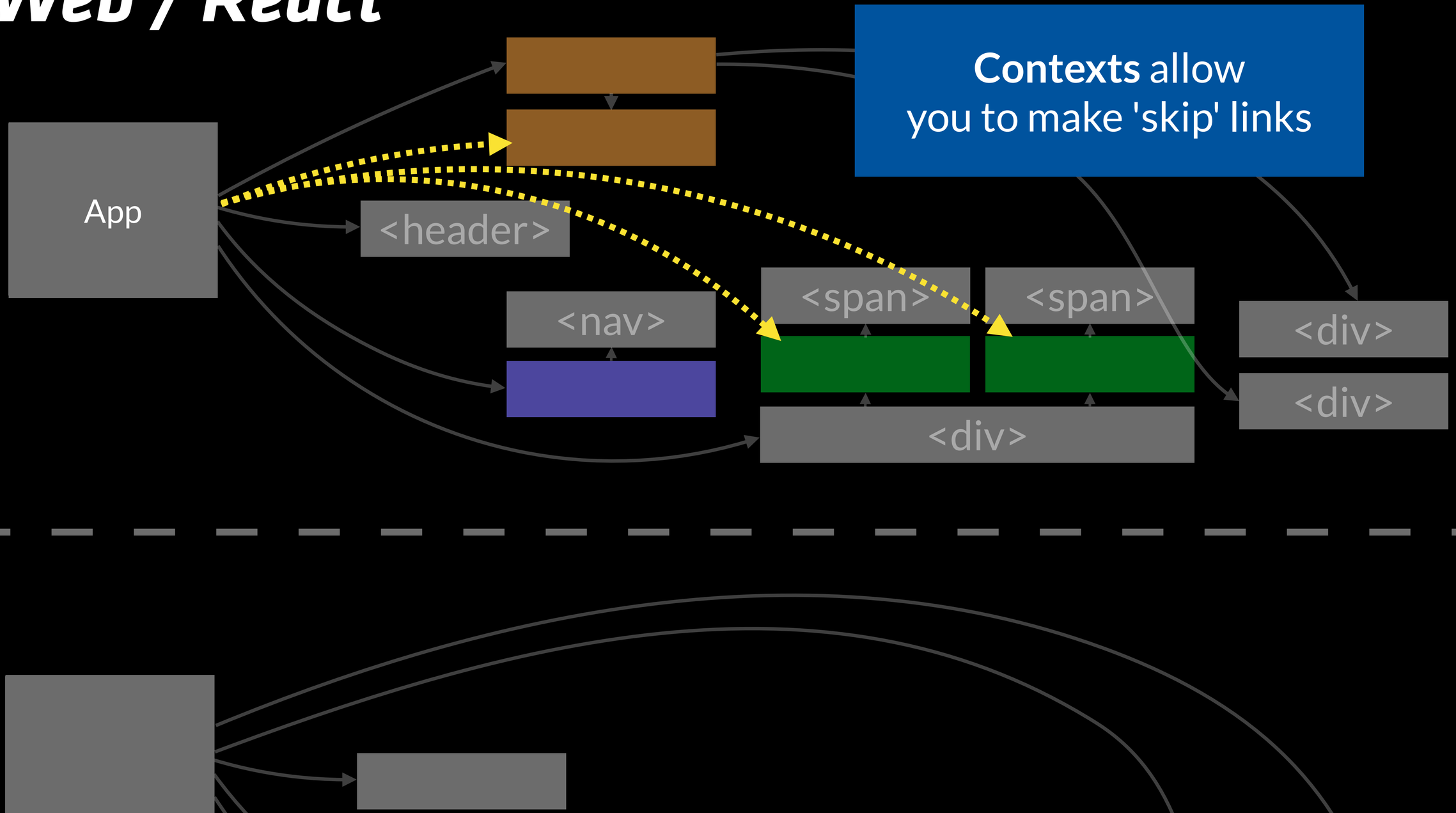




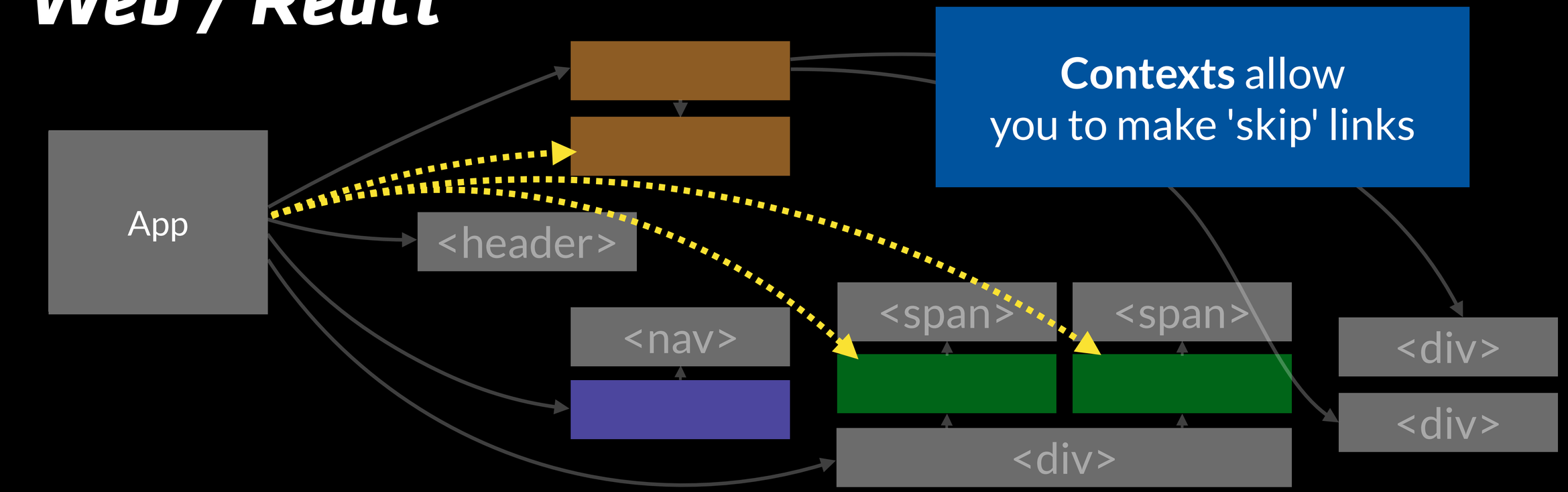
Web / React



Web / React



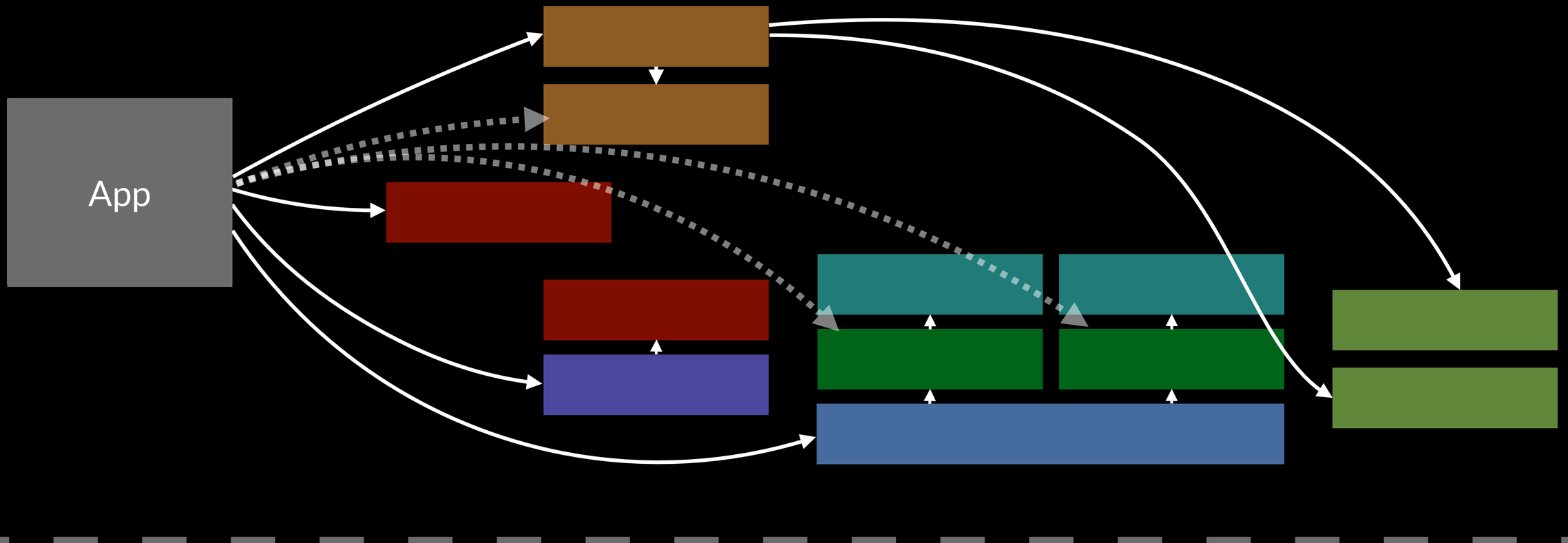
Web / React


App
<header>
<nav>

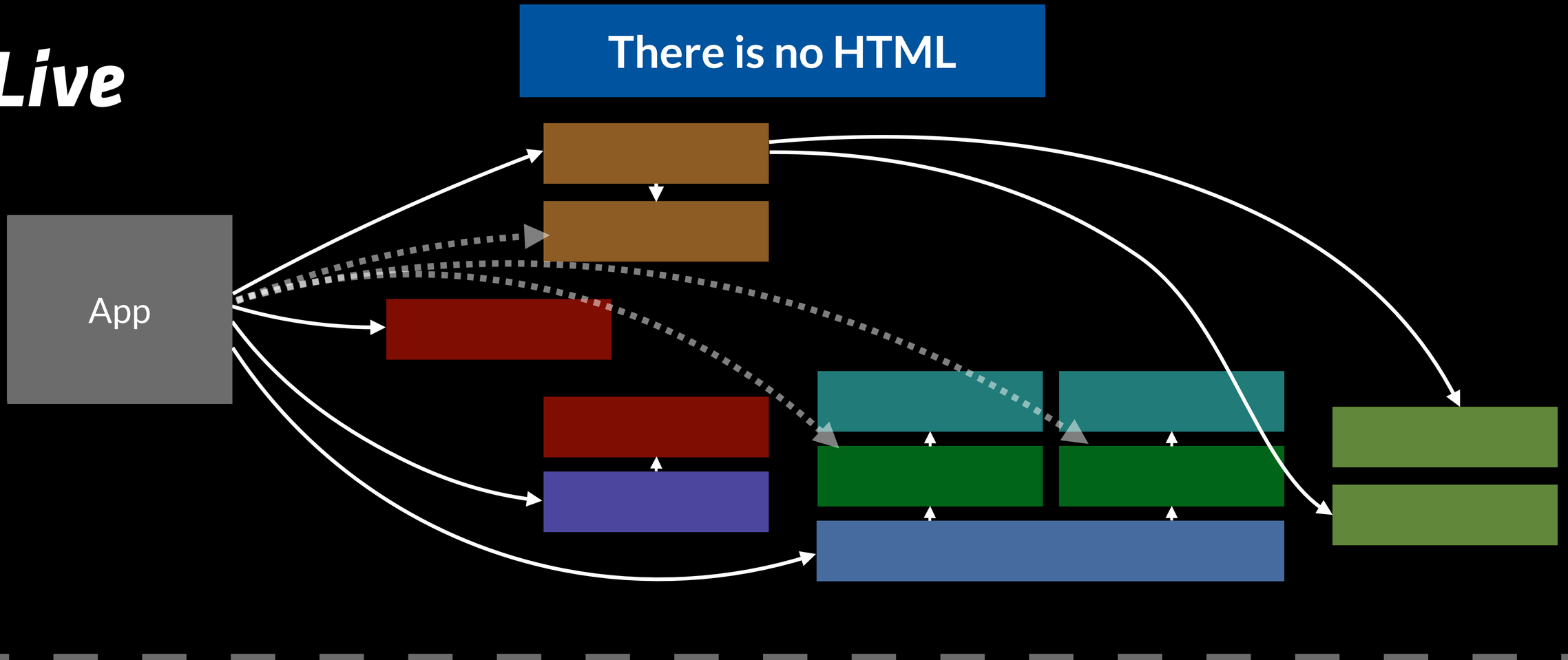
Contexts allow
you to make 'skip' links


It's a **directed graph** with a tree for a **spine**

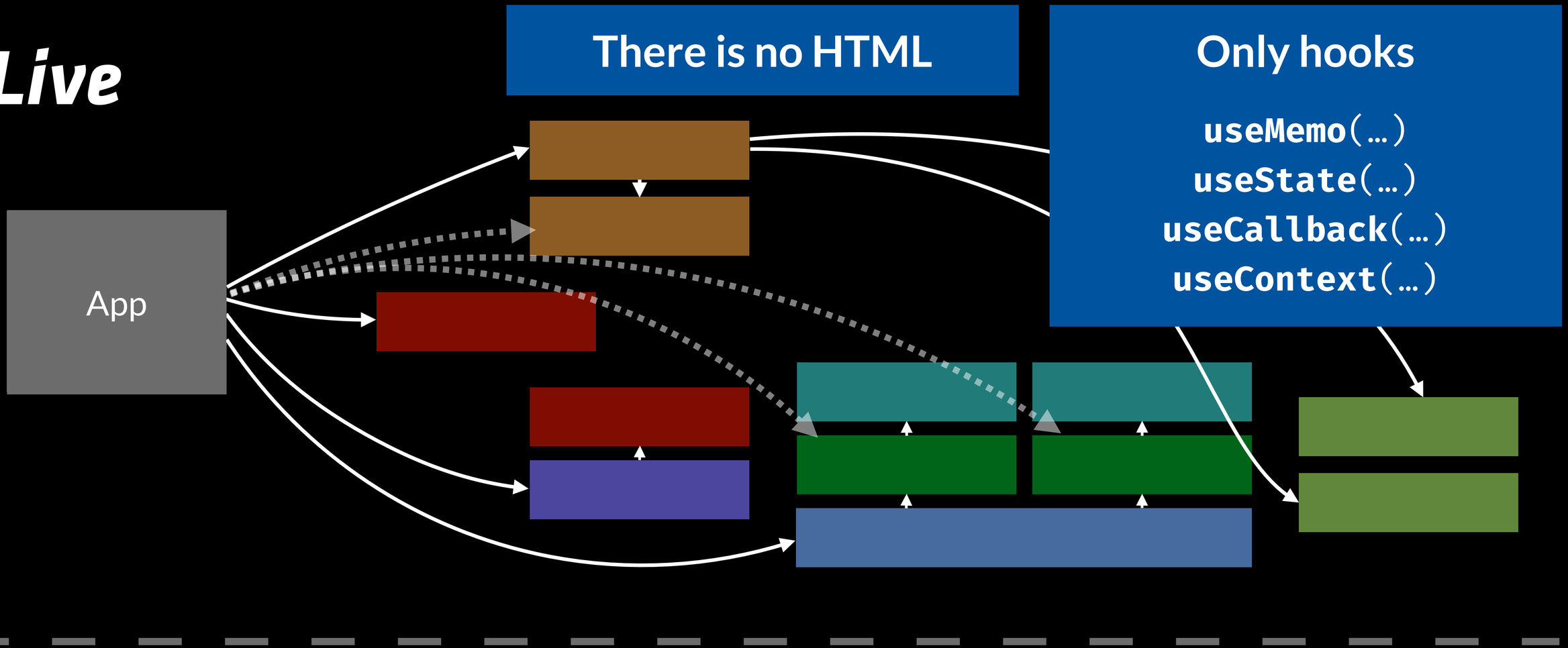
Live



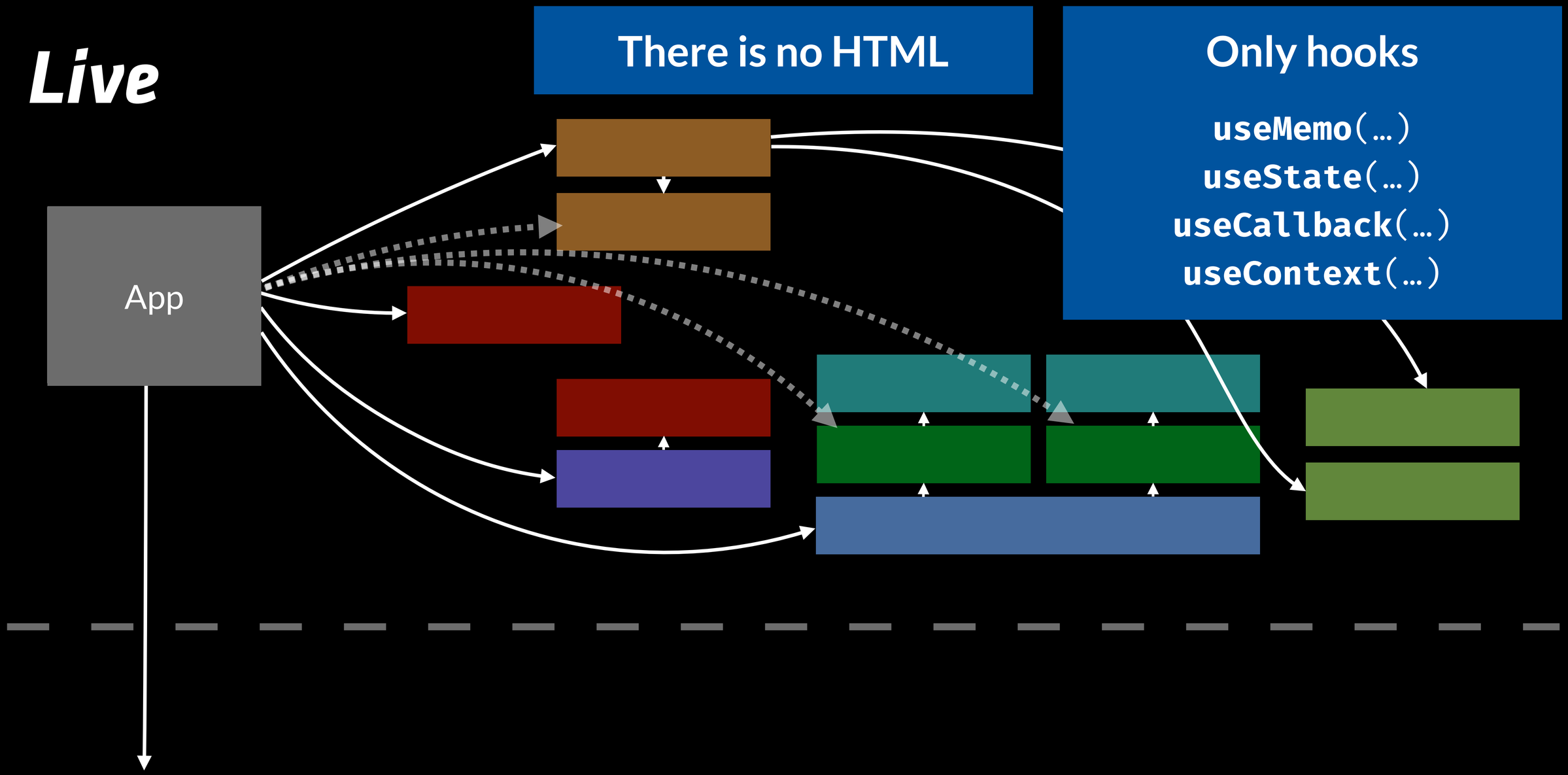
Live



Live


There is no HTML
Only hooks
useMemo(…)
useState(…)
useCallback(…)
useContext(…)
App

Live


There is no HTML
Only hooks
useMemo(…)
useState(…)
useCallback(…)
useContext(…)
App

Live

There is no HTML

Only hooks

```
useMemo(…)
useState(…)
useCallback(…)
useContext(…)
```

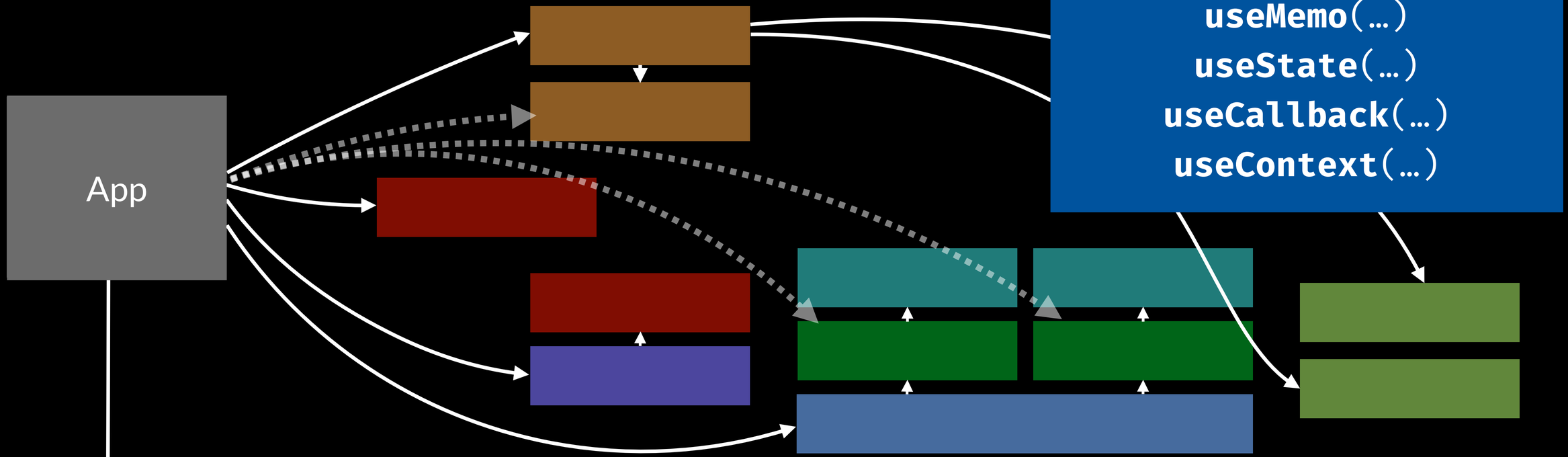
Live drives a **tree** just like the first

Live

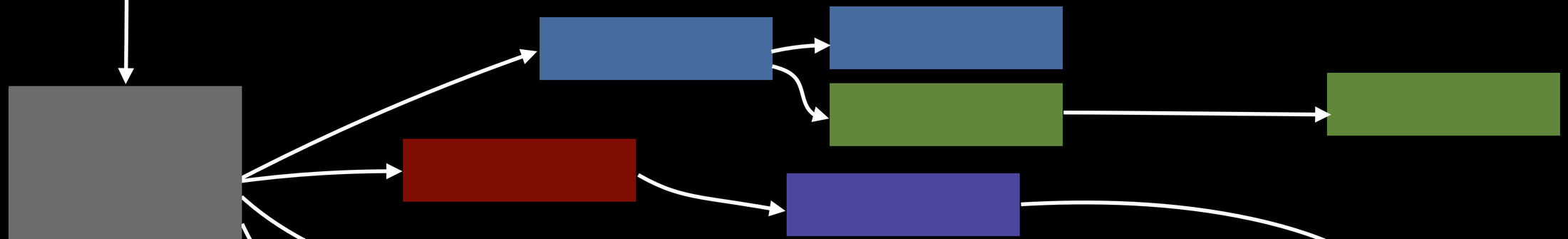
There is no HTML

Only hooks

`useMemo(…)`
`useState(…)`
`useCallback(…)`
`useContext(…)`



Live drives a **tree** just like the first

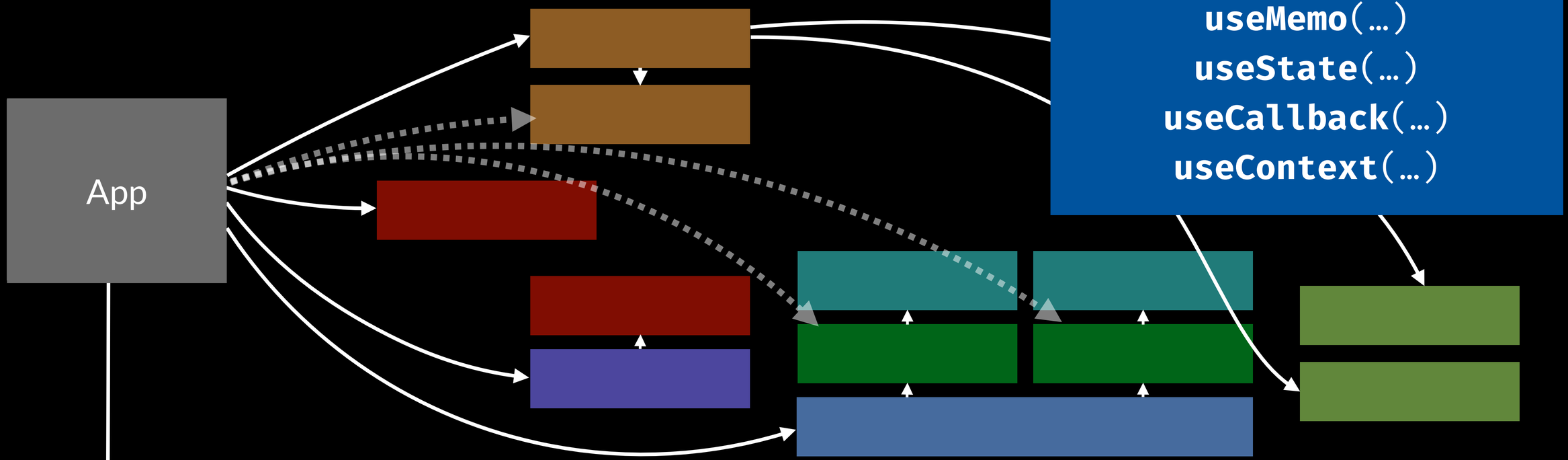


Live

There is no HTML

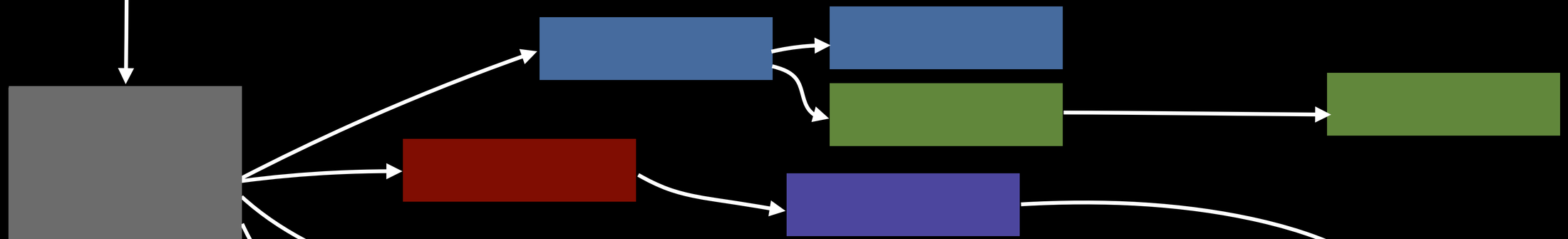
Only hooks

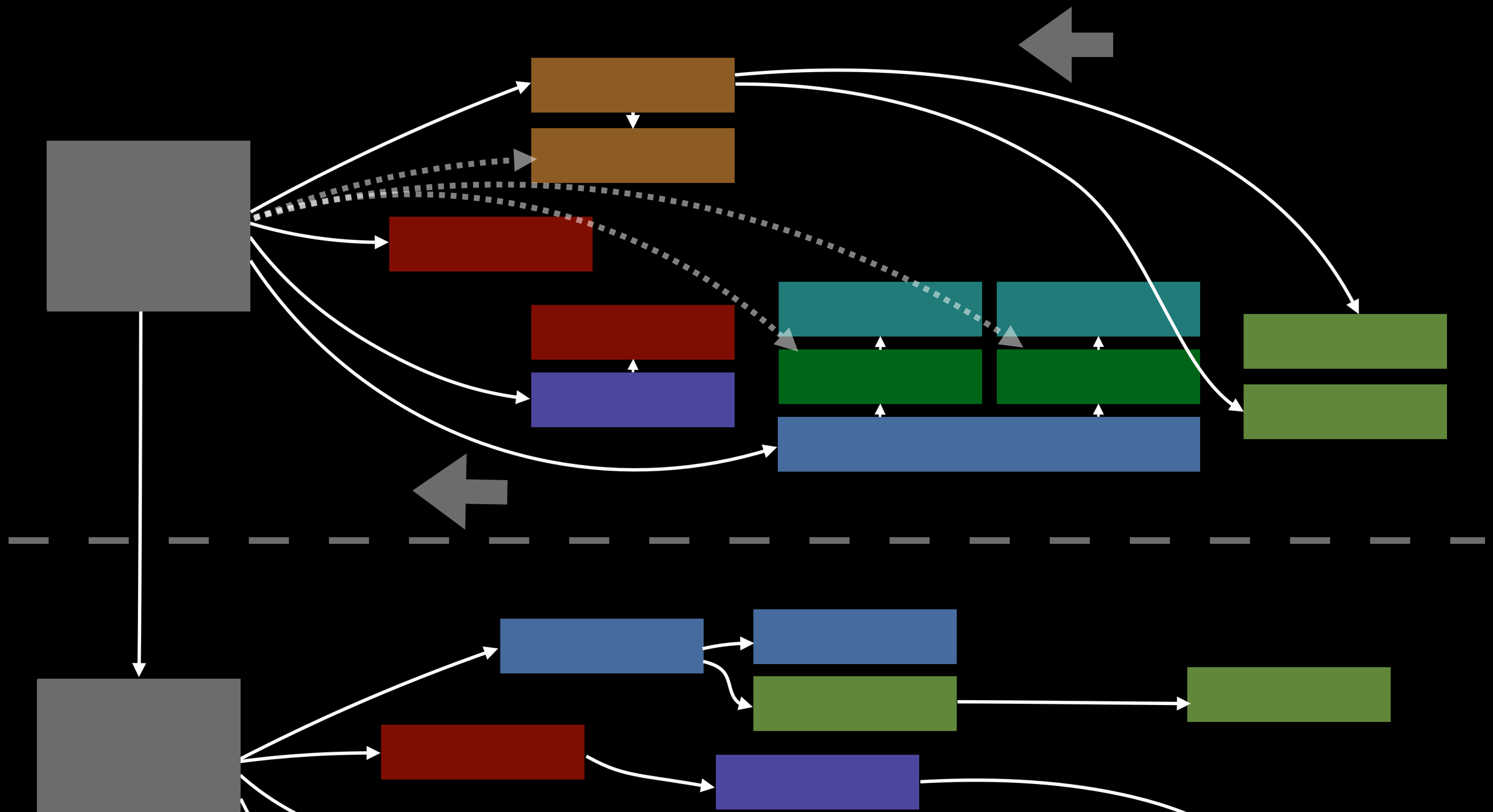
`useMemo(…)`
`useState(…)`
`useCallback(…)`
`useContext(…)`

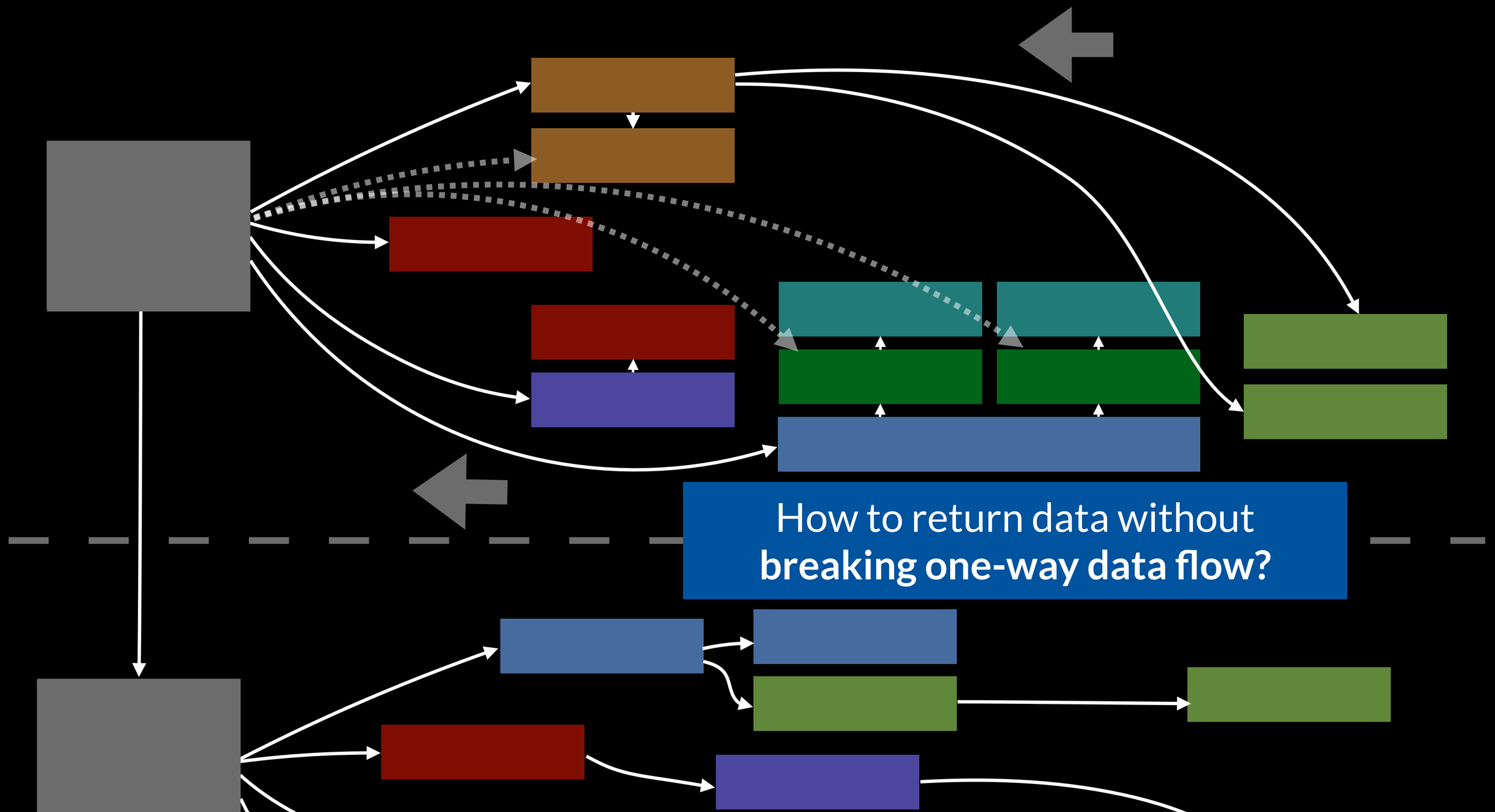


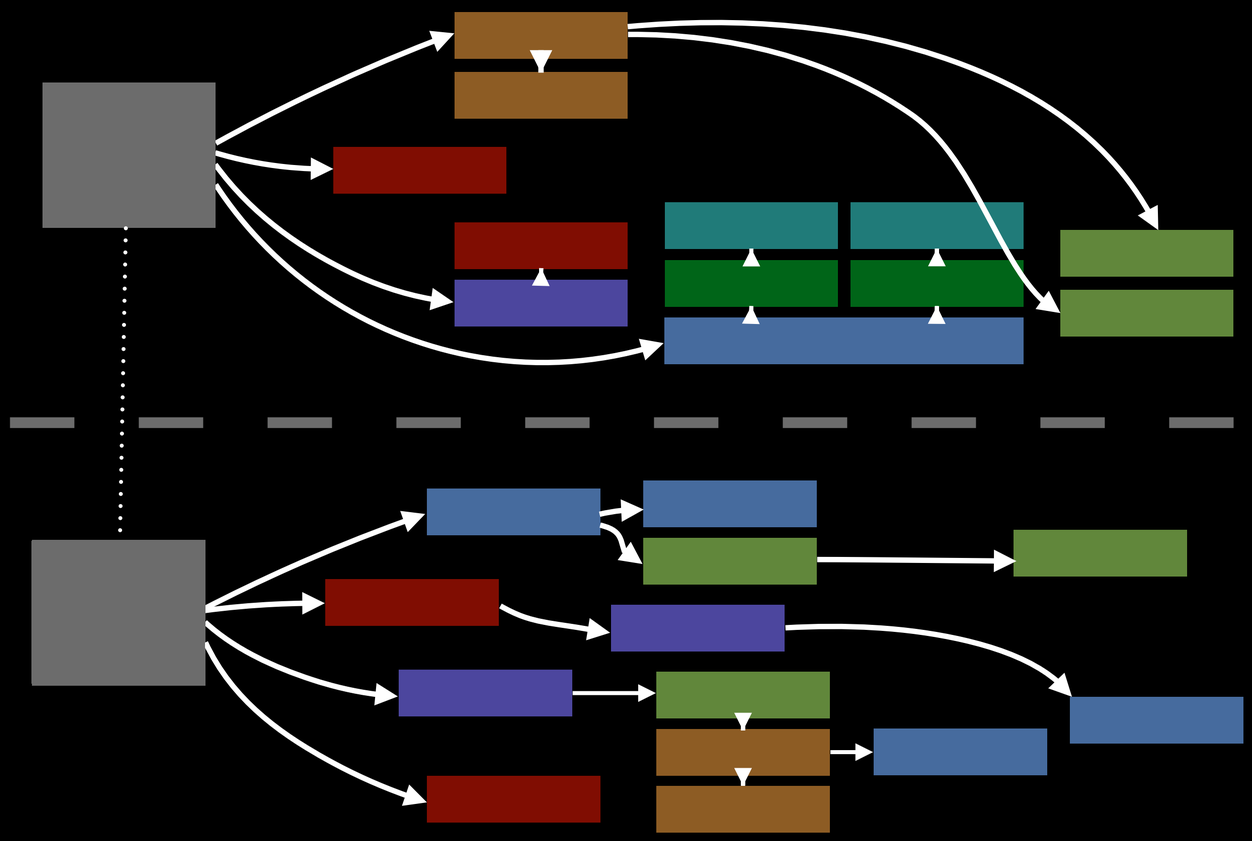
Live drives a **tree** just like the first

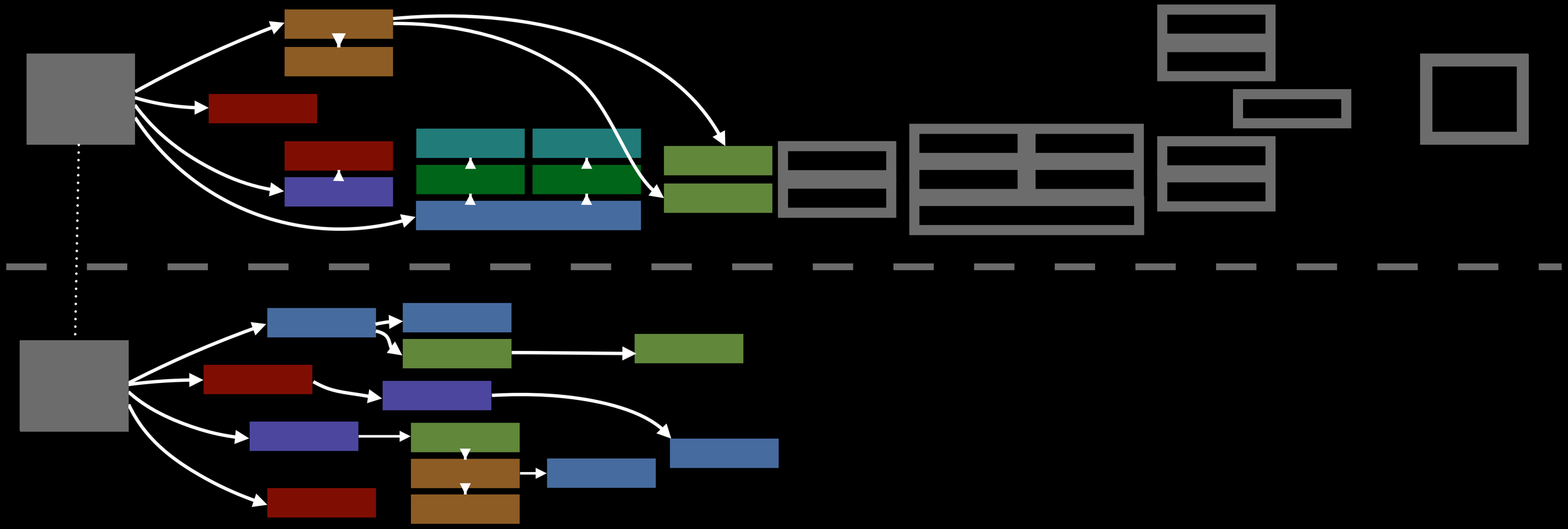
But they don't **have to** match



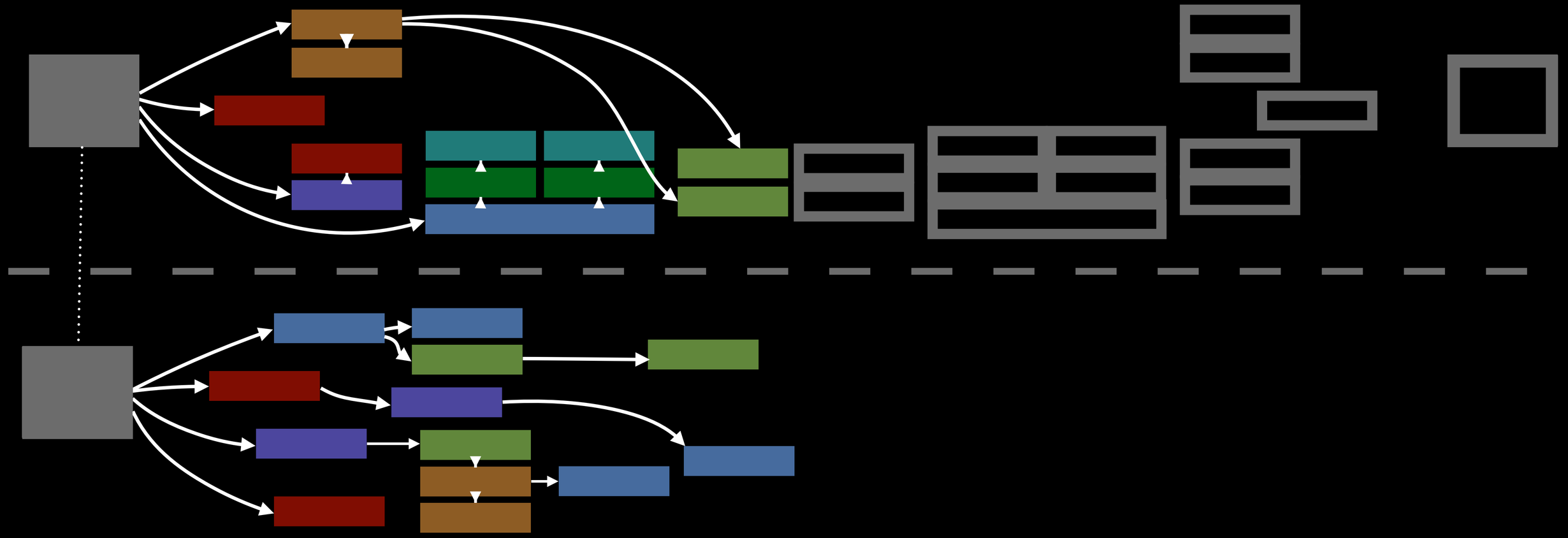



How to return data without
breaking one-way data flow?

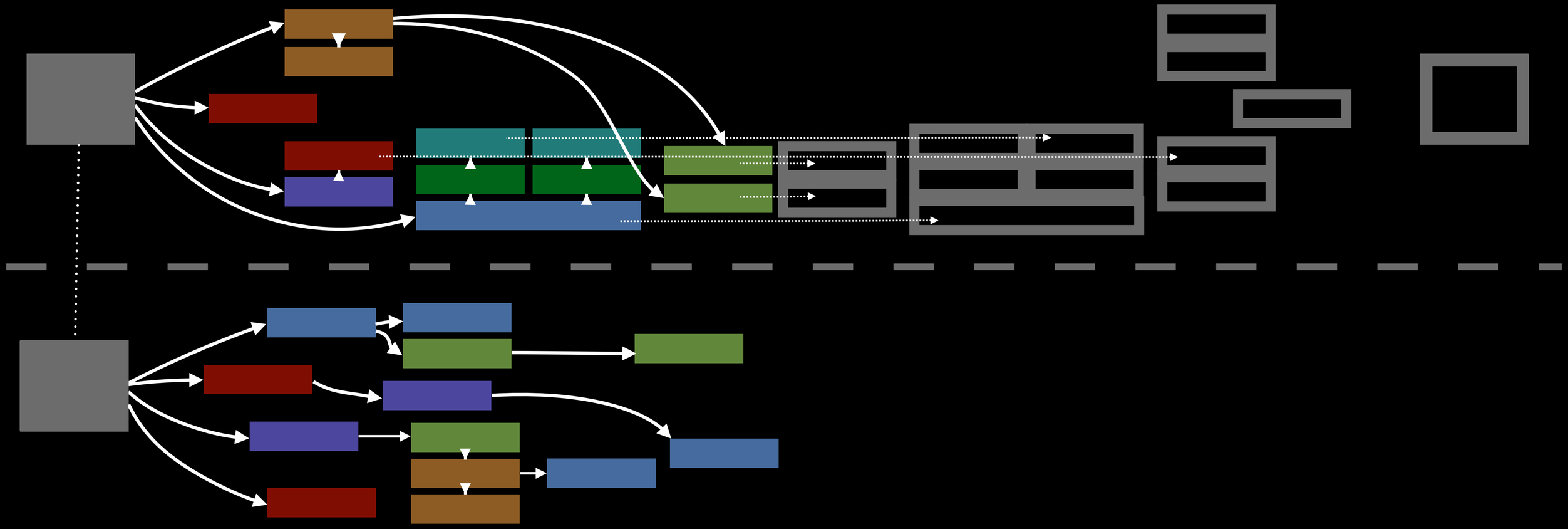




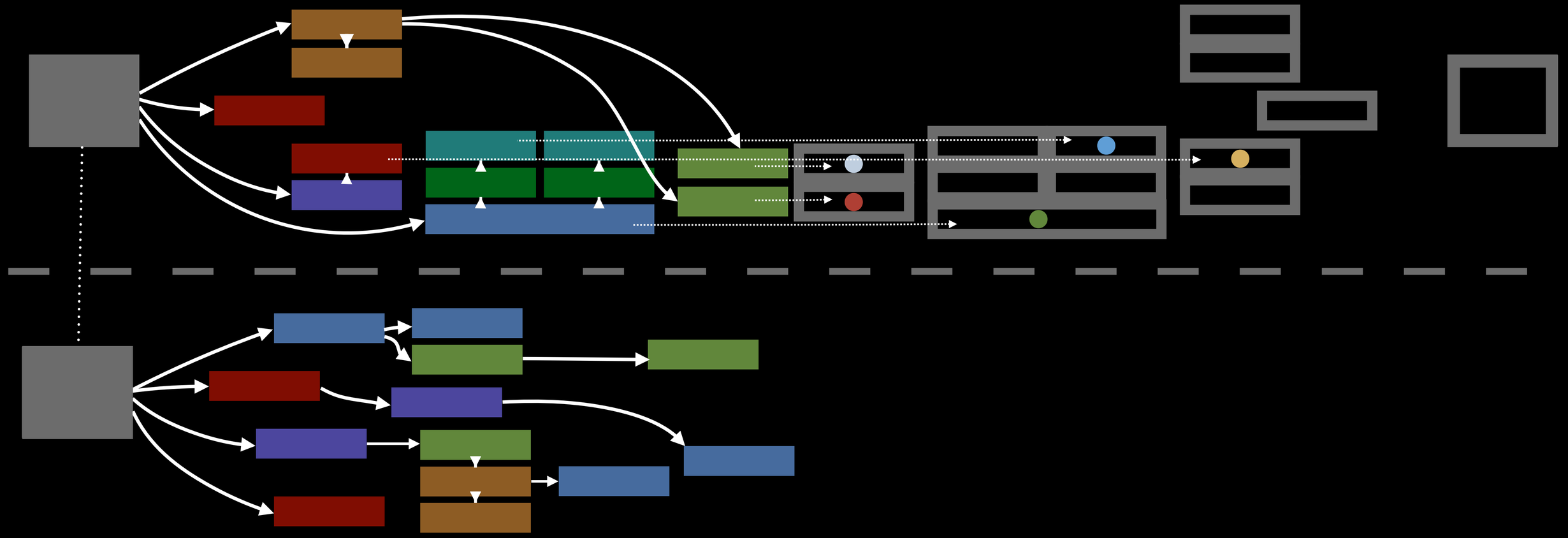
Reverse shadow tree

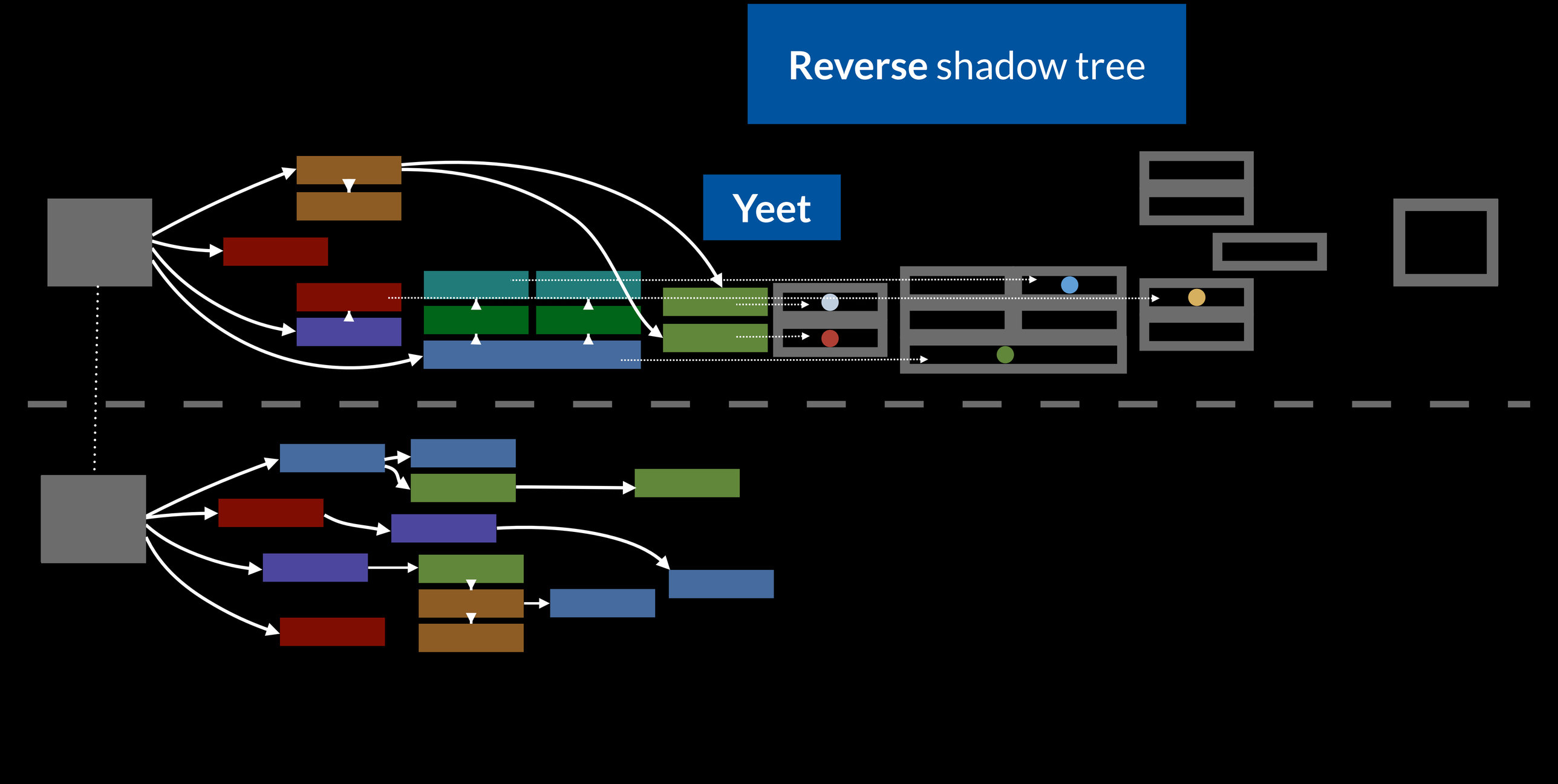


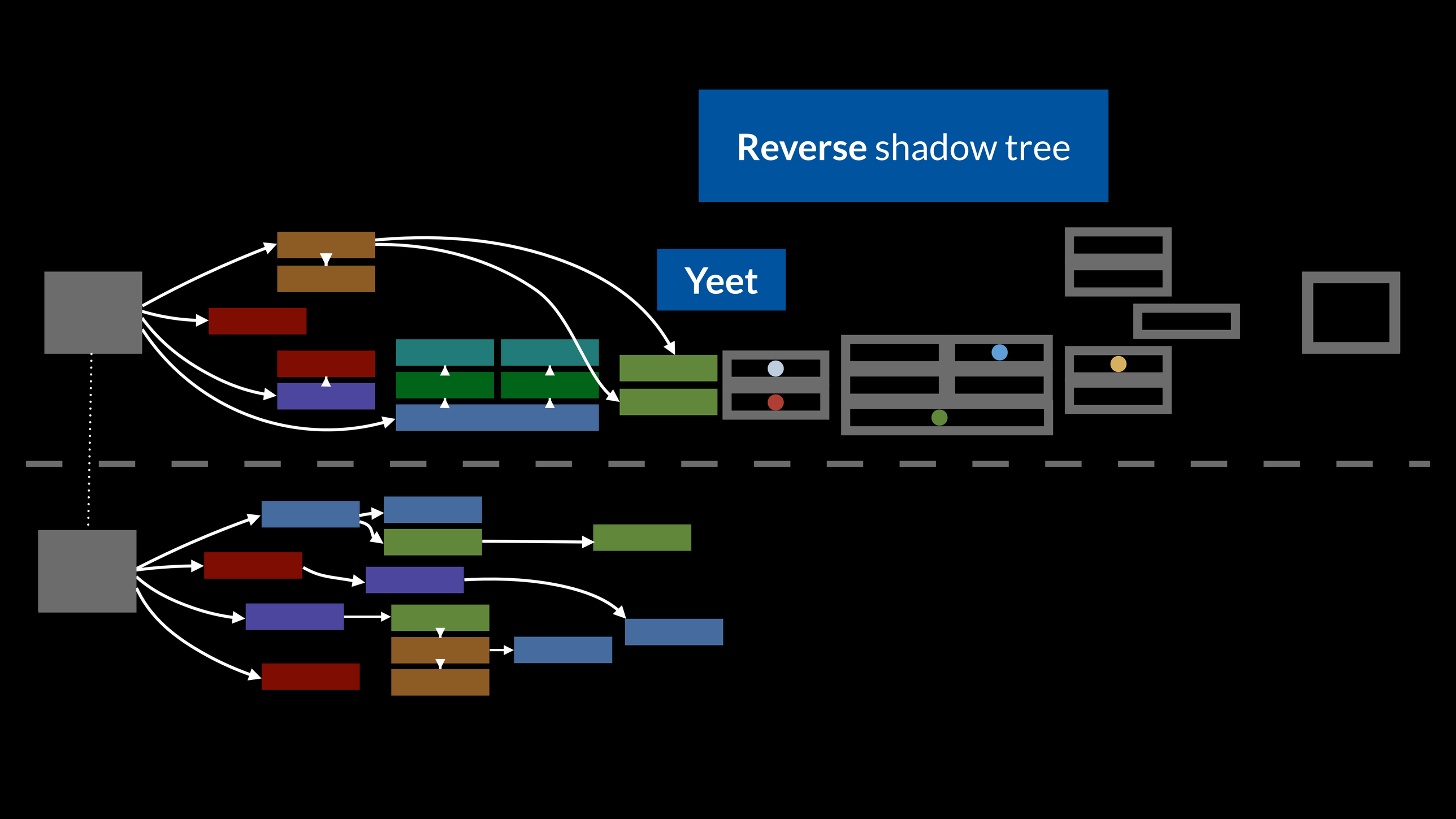
Reverse shadow tree



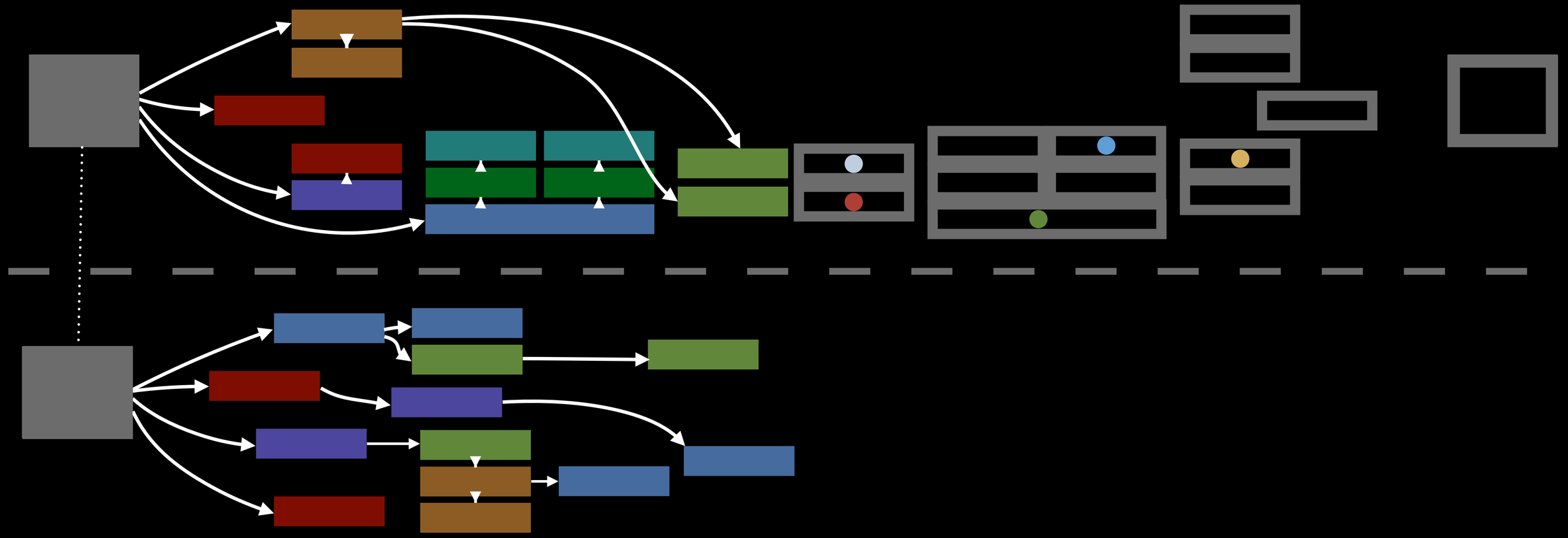
Reverse shadow tree



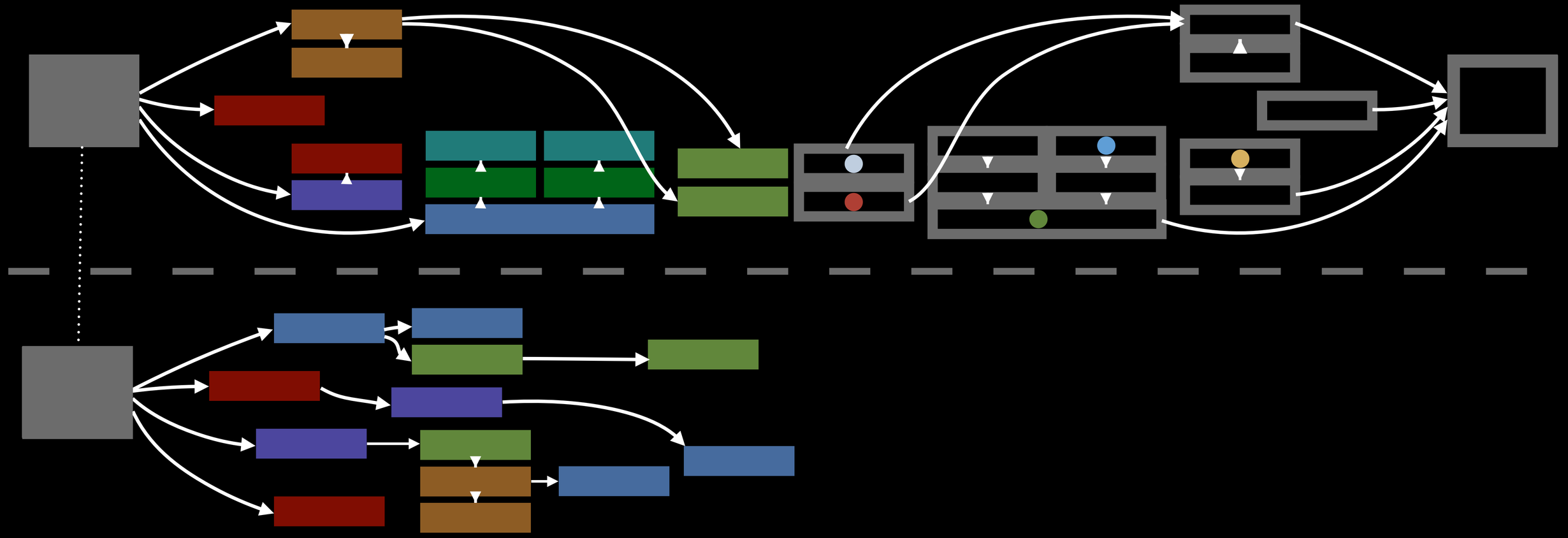

Reverse shadow tree
Yeet


Reverse shadow tree
Yeet

Reverse shadow tree

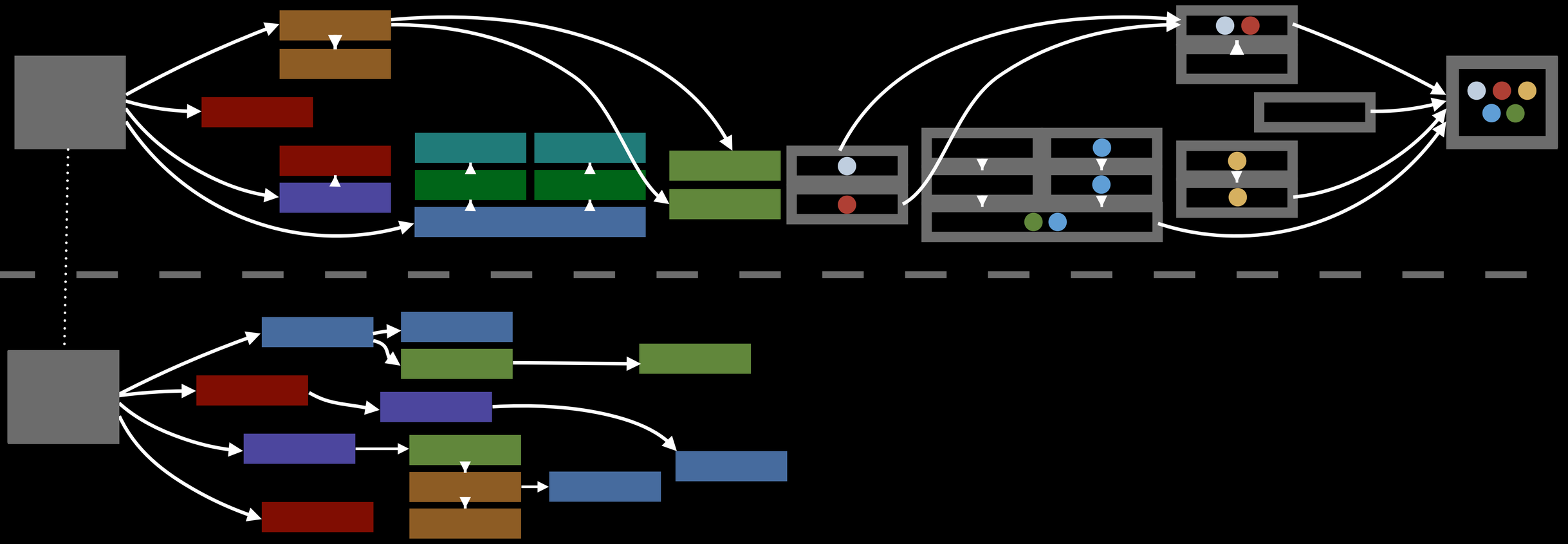


Reverse shadow tree



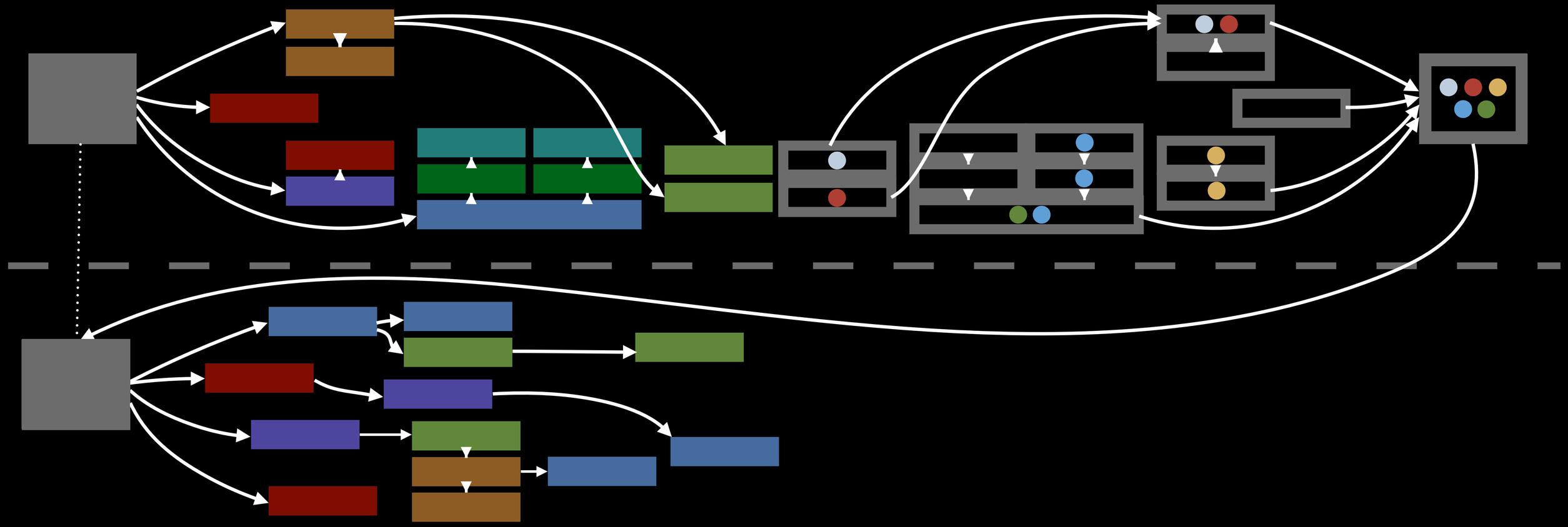
Reverse shadow tree

Incremental Map Reduce



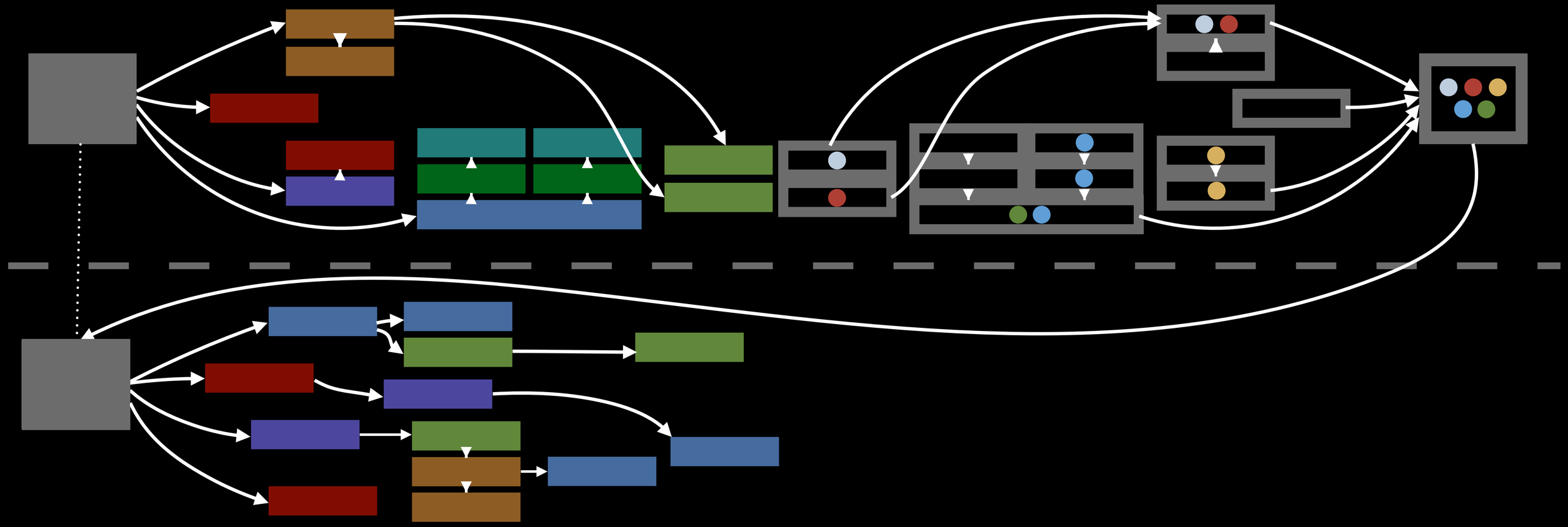
Reverse shadow tree

Incremental Map Reduce

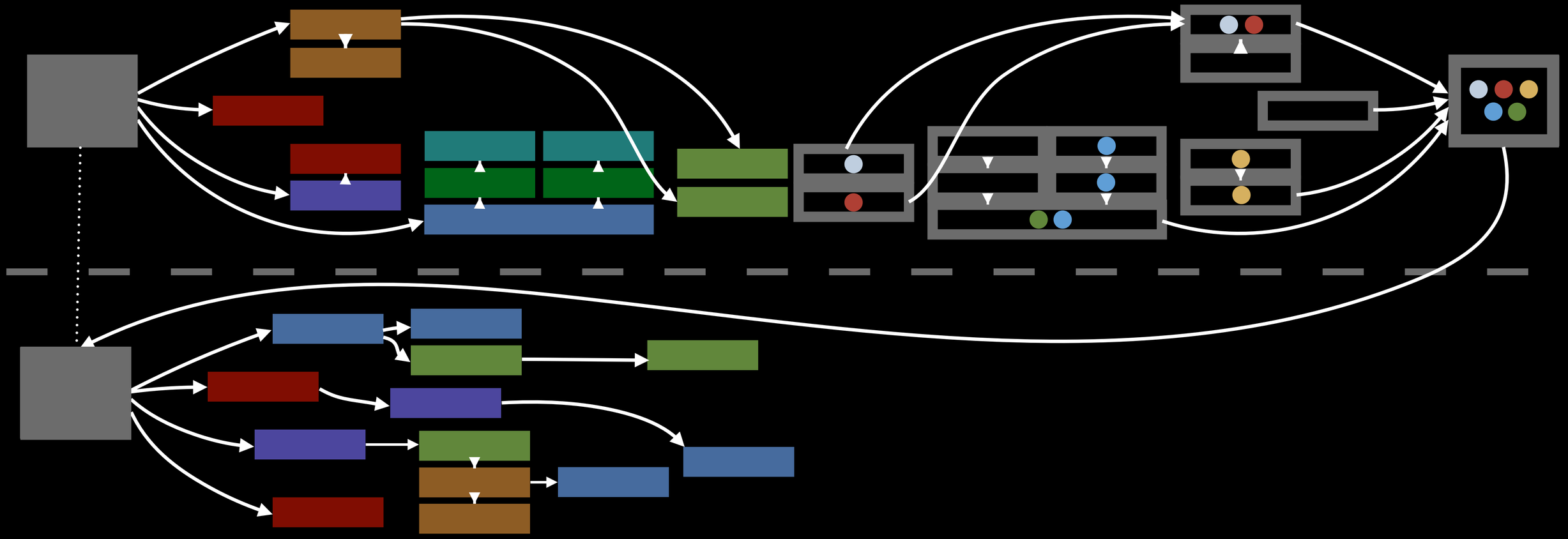


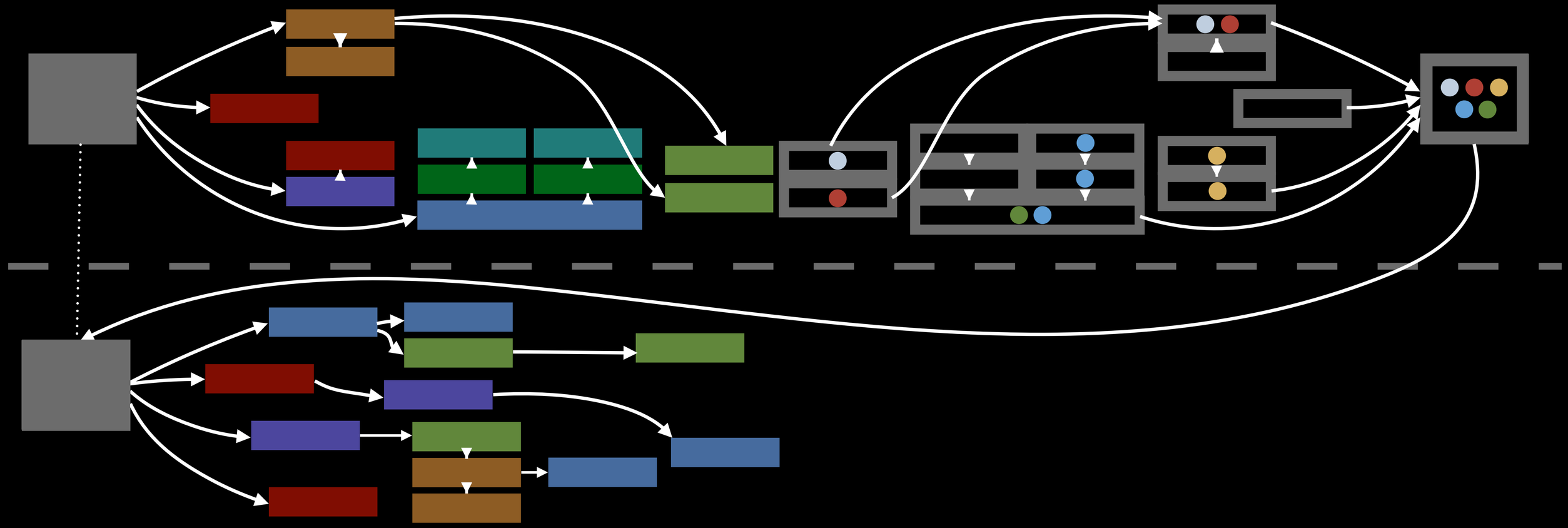
Reverse shadow tree

Incremental Map Reduce

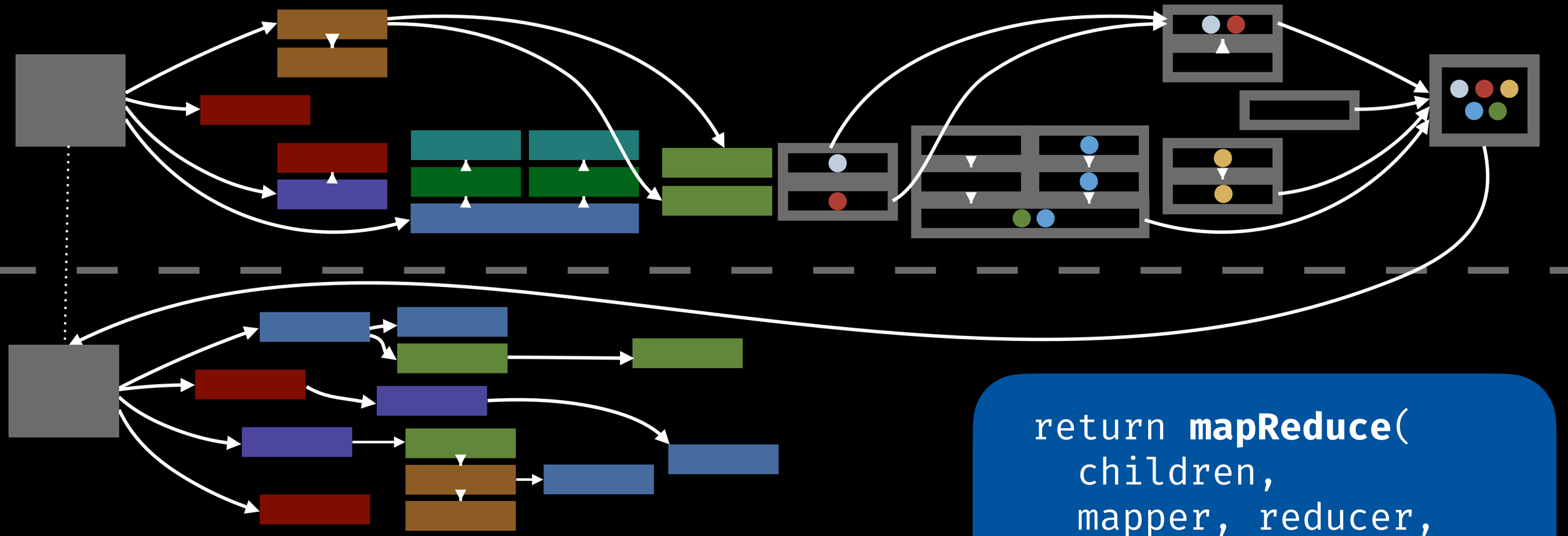


Continuation



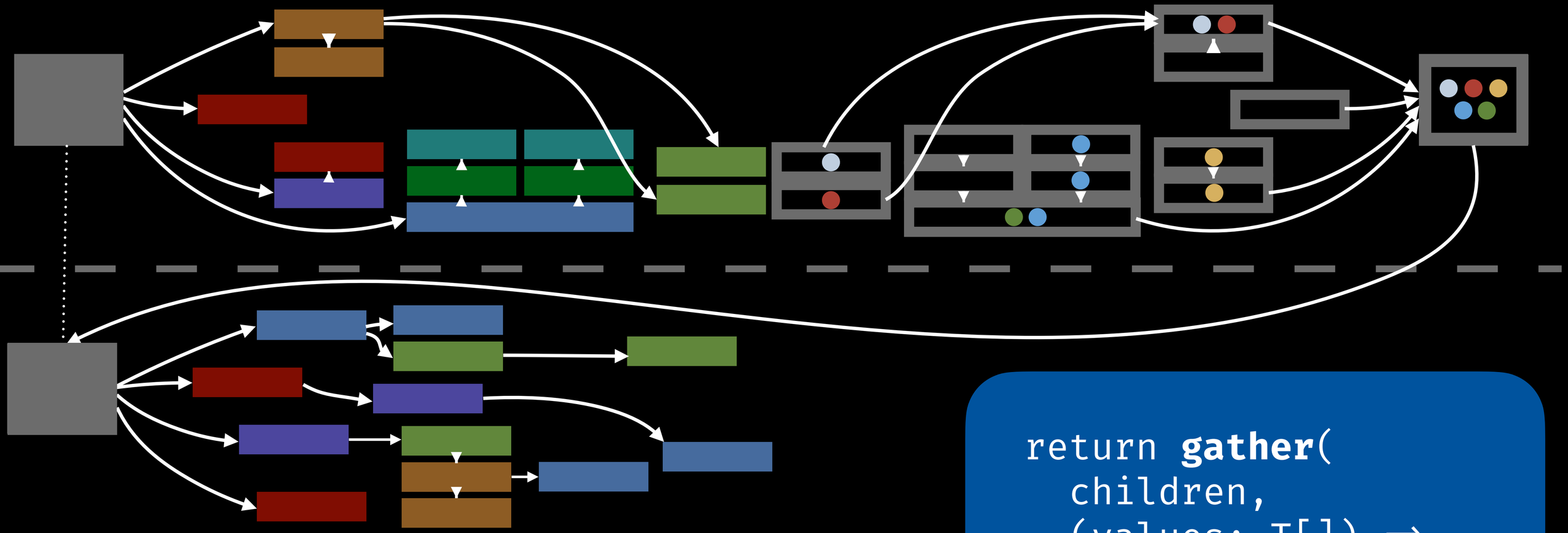


Incremental version of **`await`**



Incremental version of **`await`**

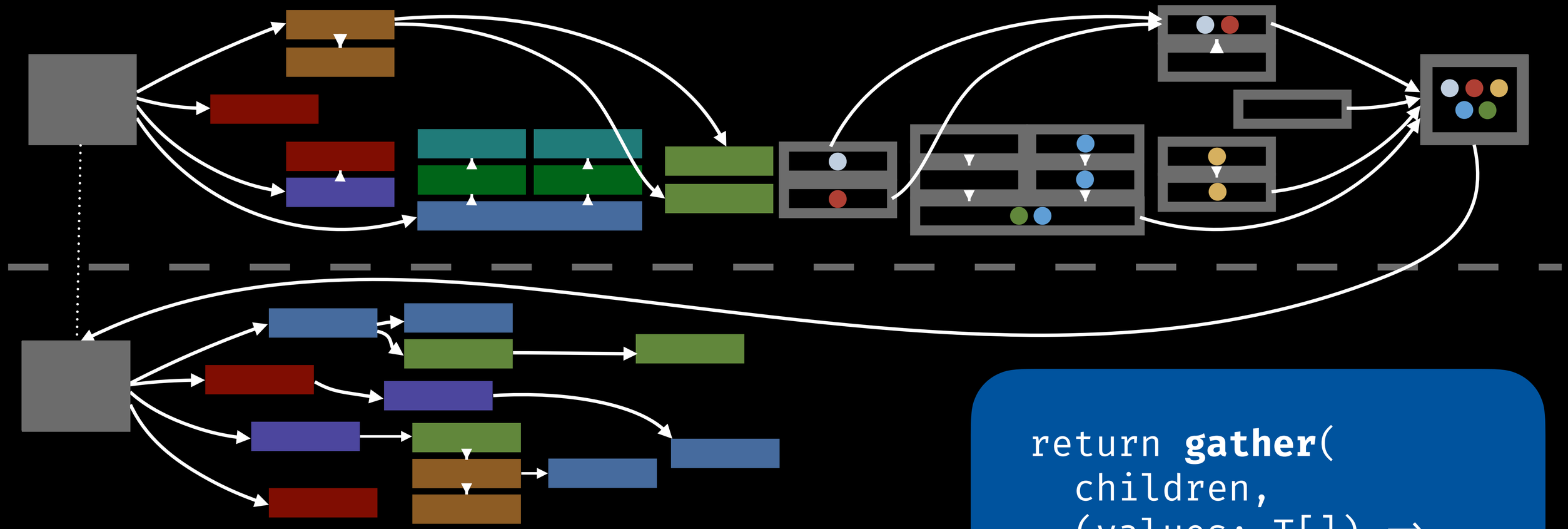
```
return mapReduce(
  children,
  mapper, reducer,
  (values: T) ⇒ …
)
```



```
return gather(
  children,
  (values: T[]) ⇒ …
)
```

Incremental version of **`await`**

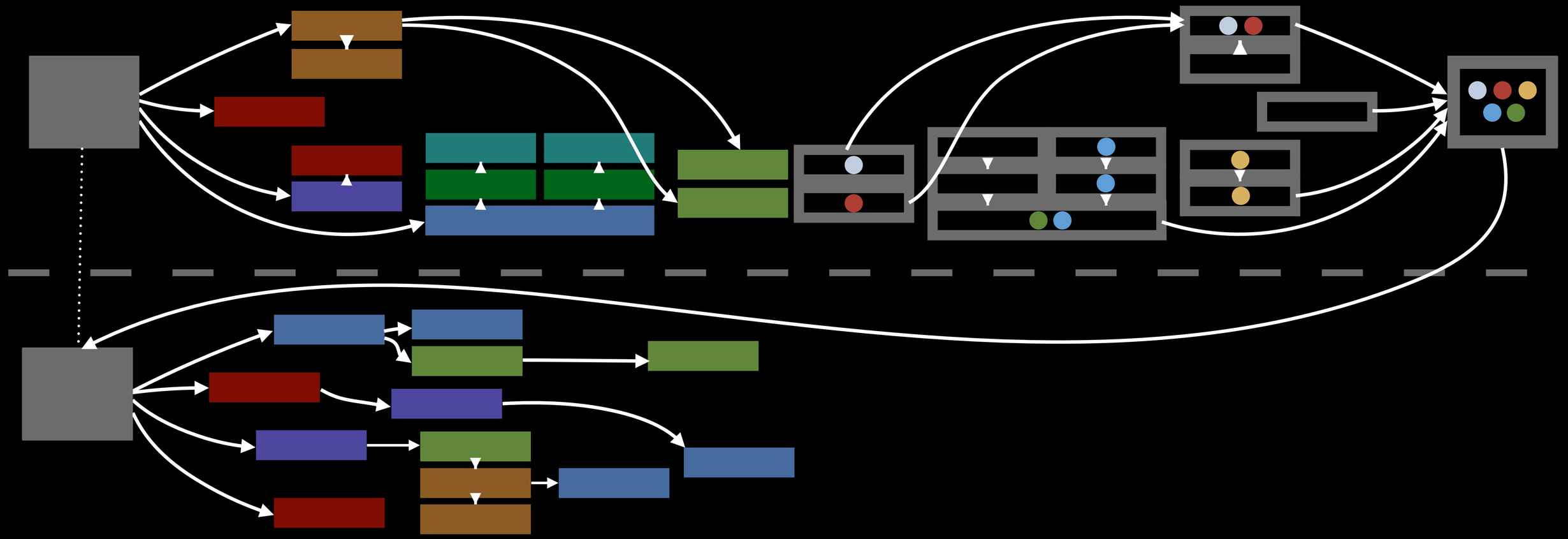
Ugly continuation passing style needed because no **rewindable generators**



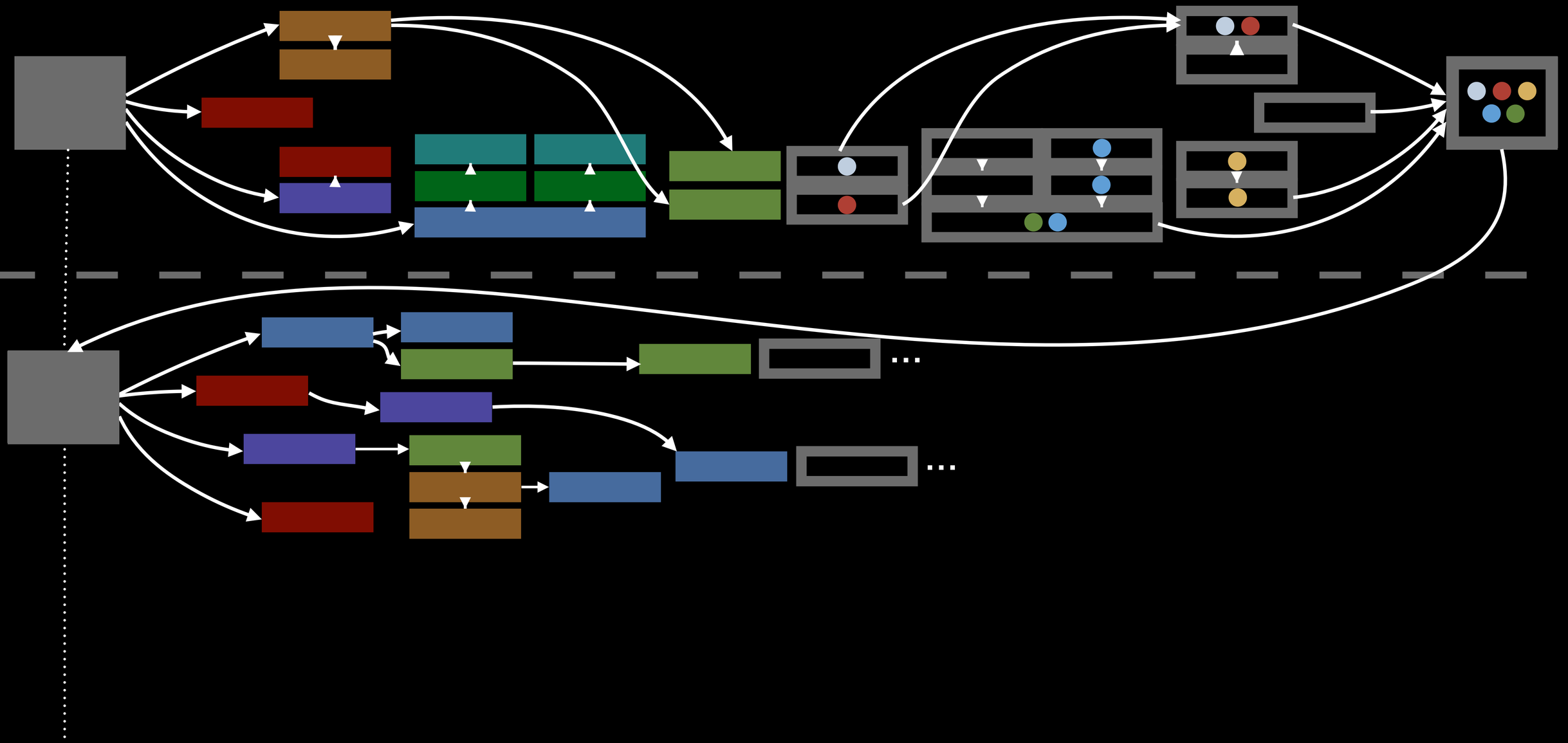
```
return gather(
  children,
  (values: T[]) ⇒ …
)
```

Incremental version of **`await`**

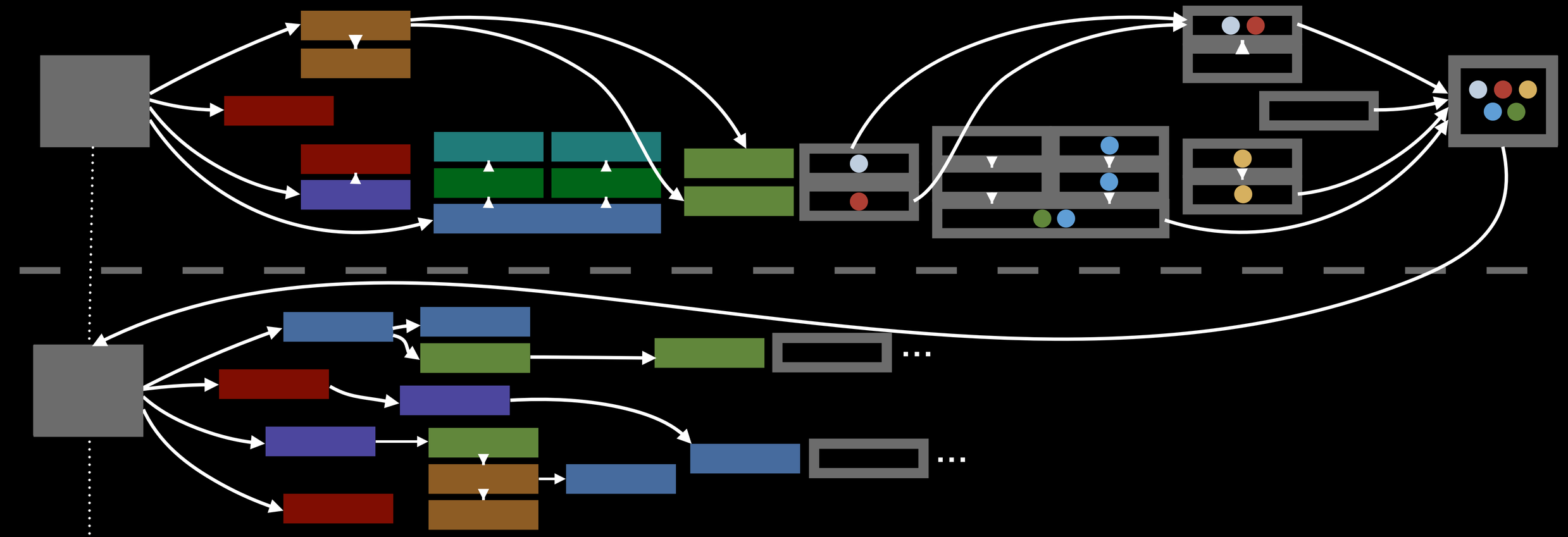
Live effect system



Live effect system

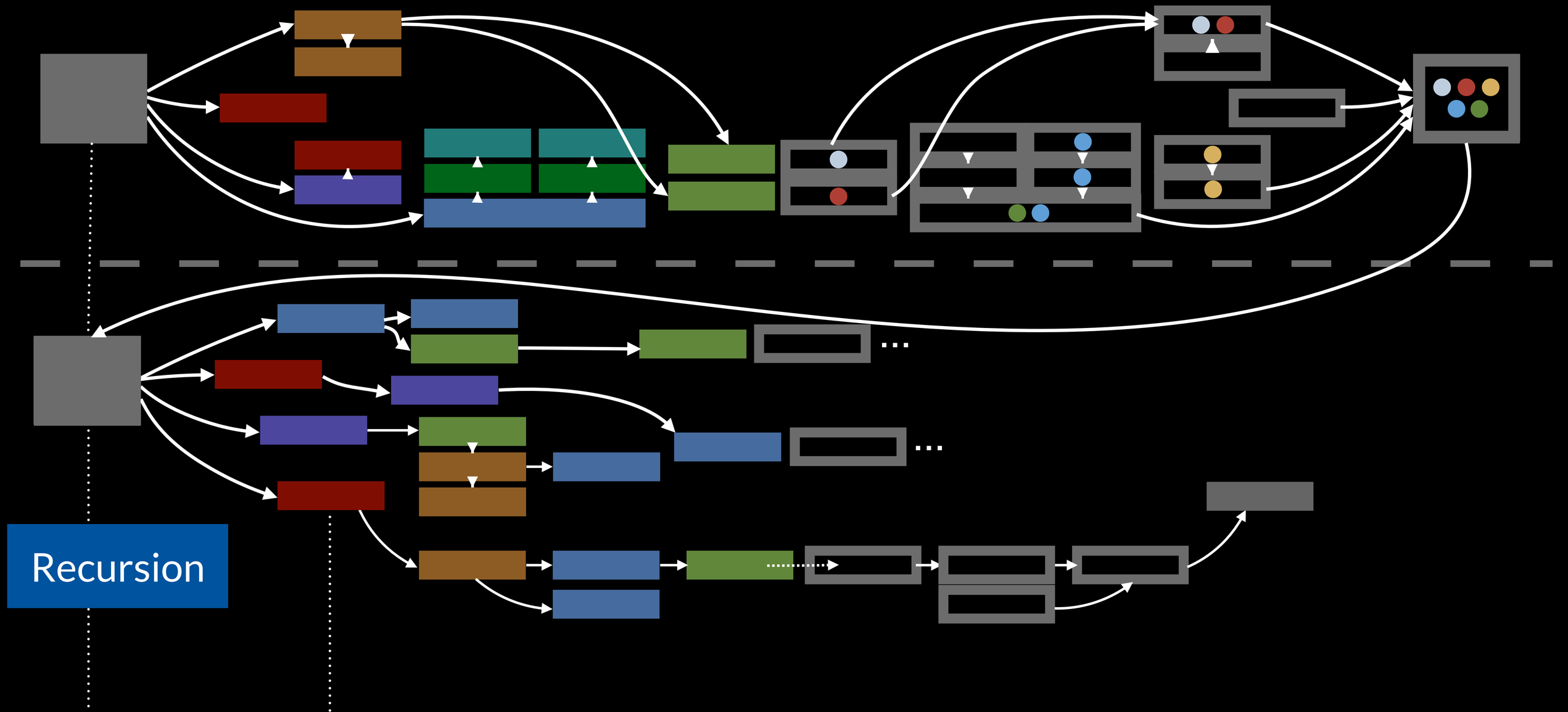


Live effect system

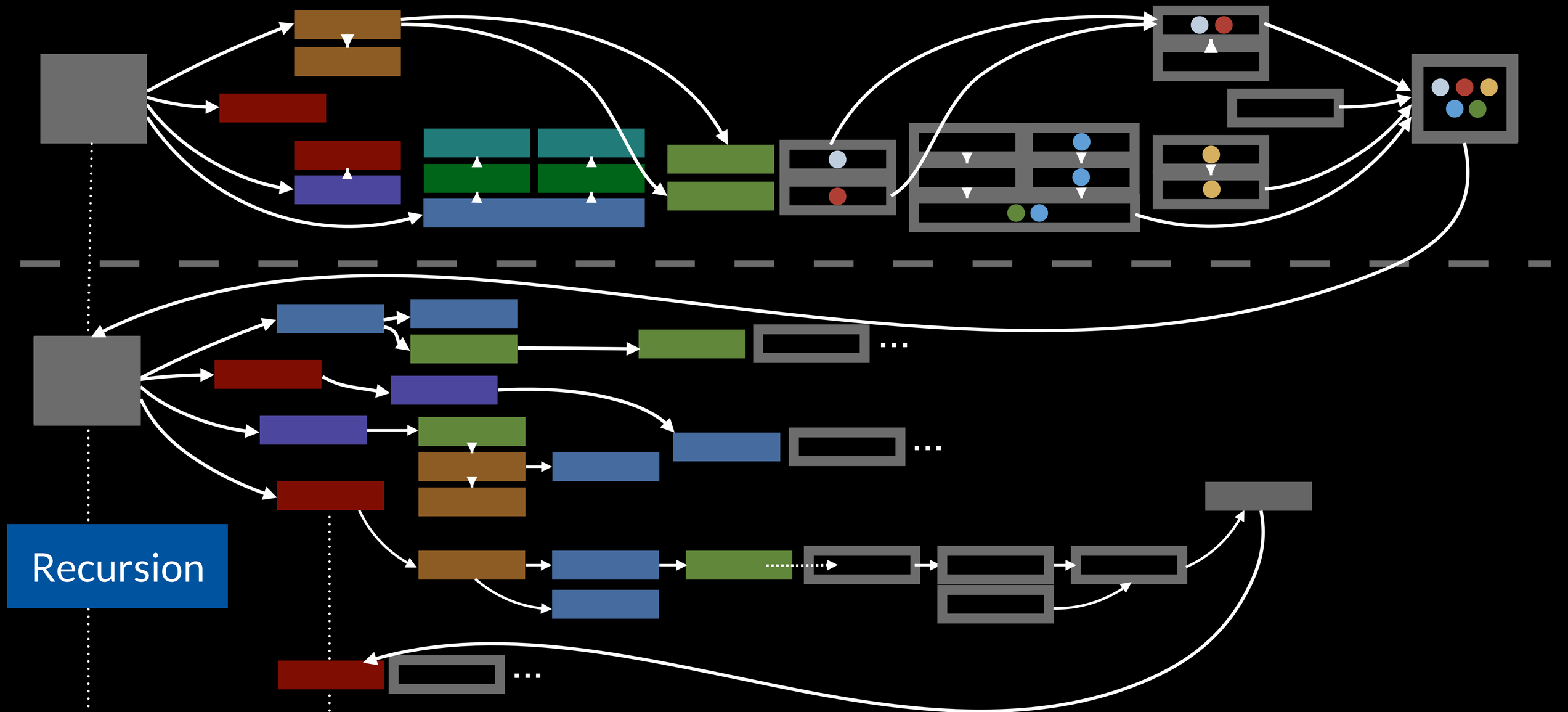


Recursion

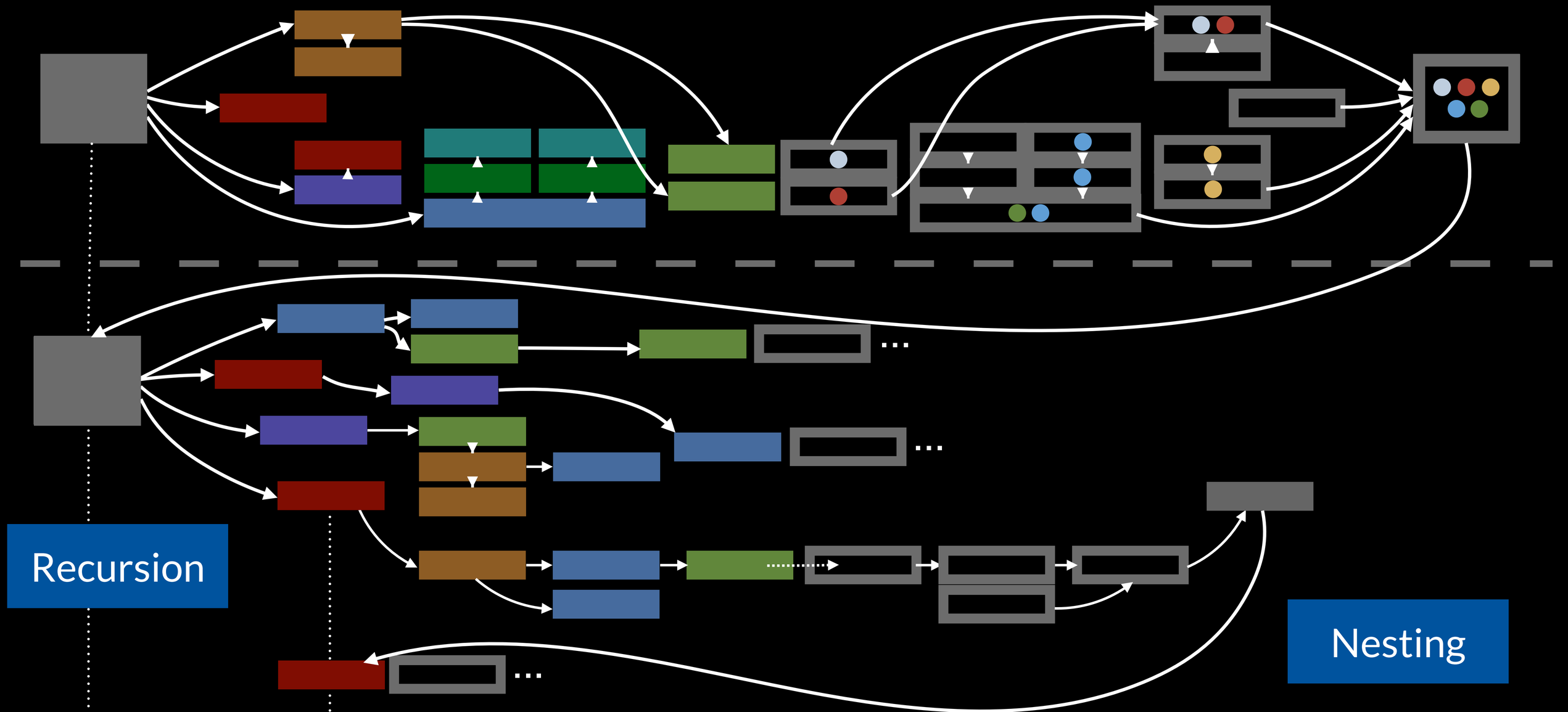
Live effect system



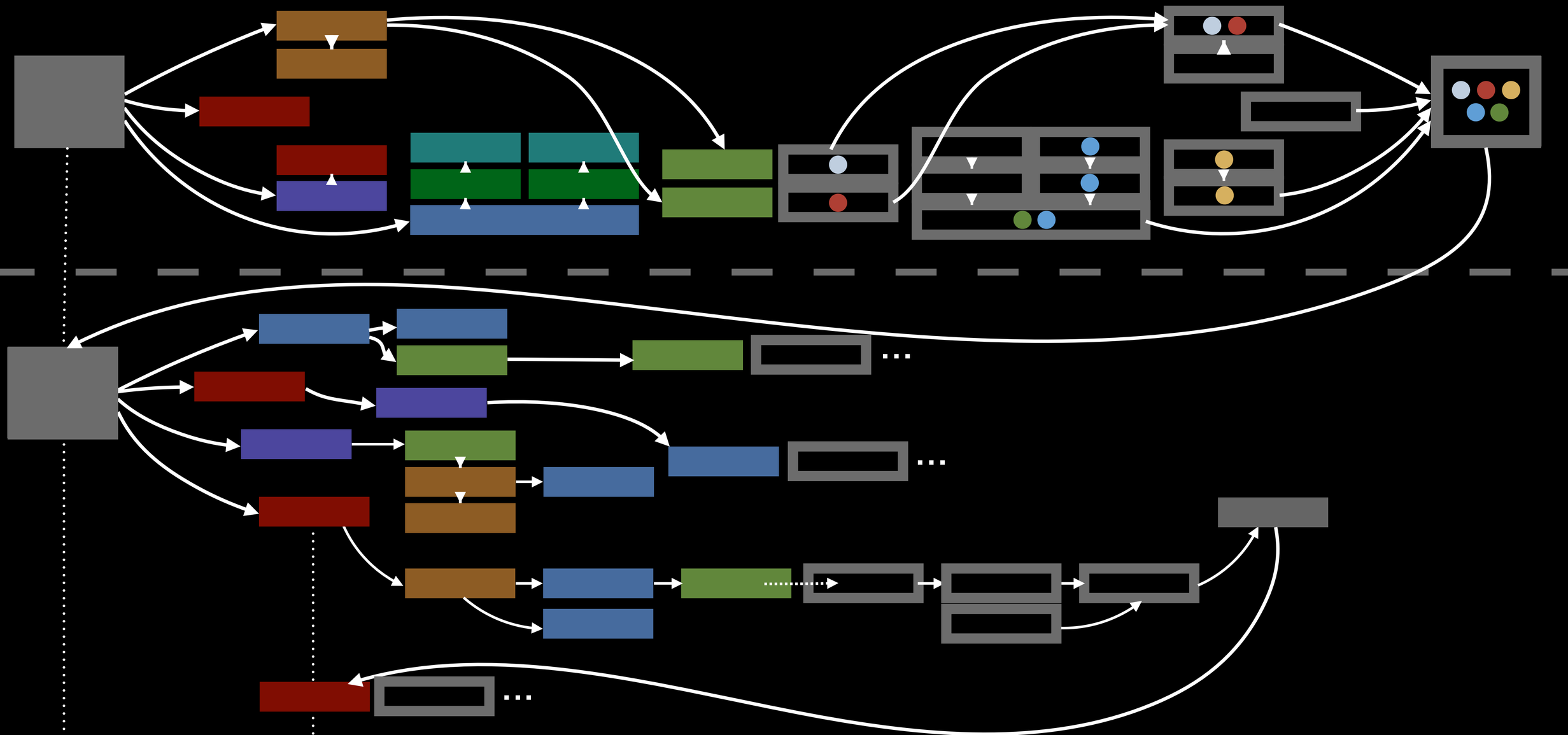
Live effect system

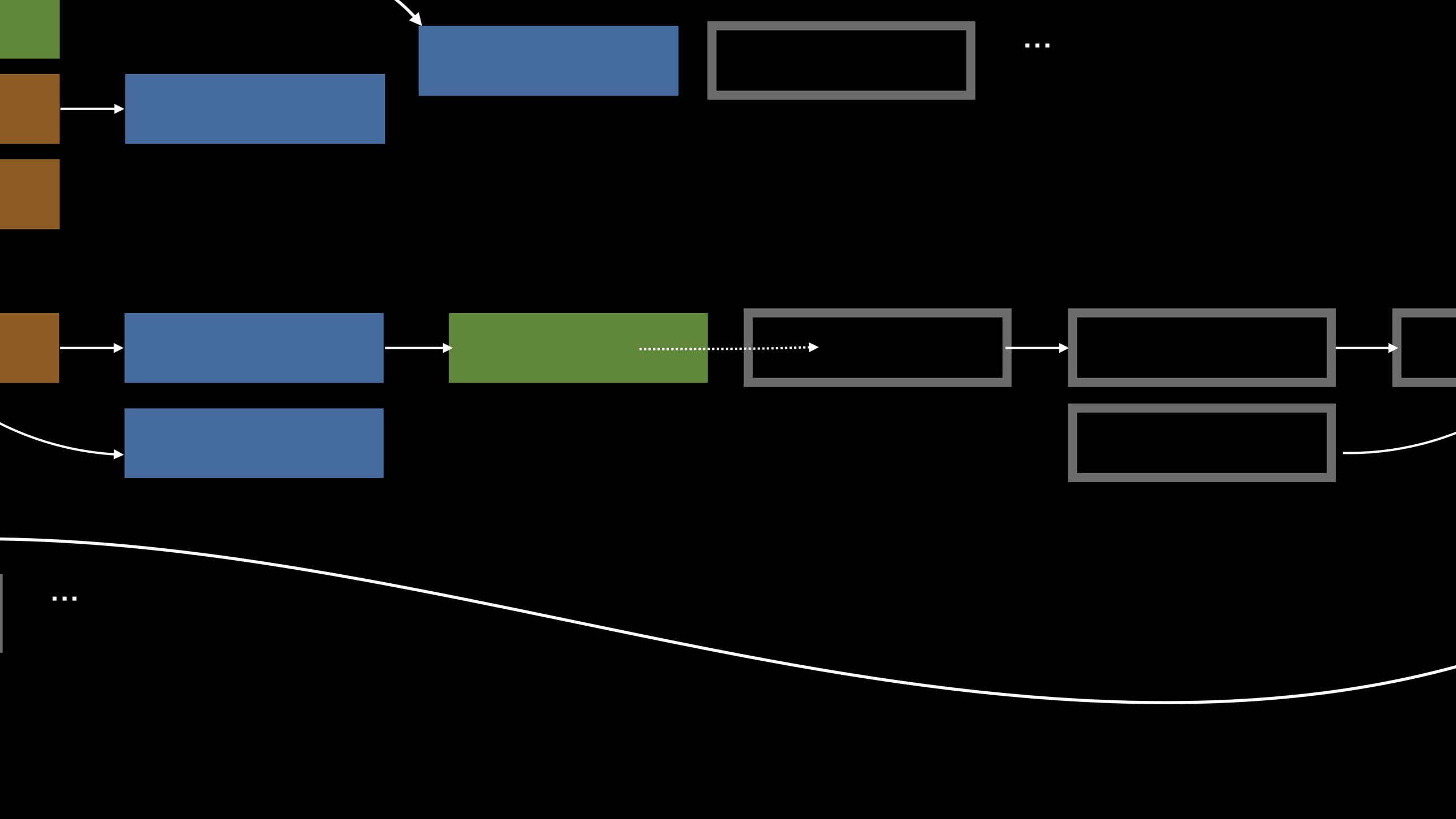


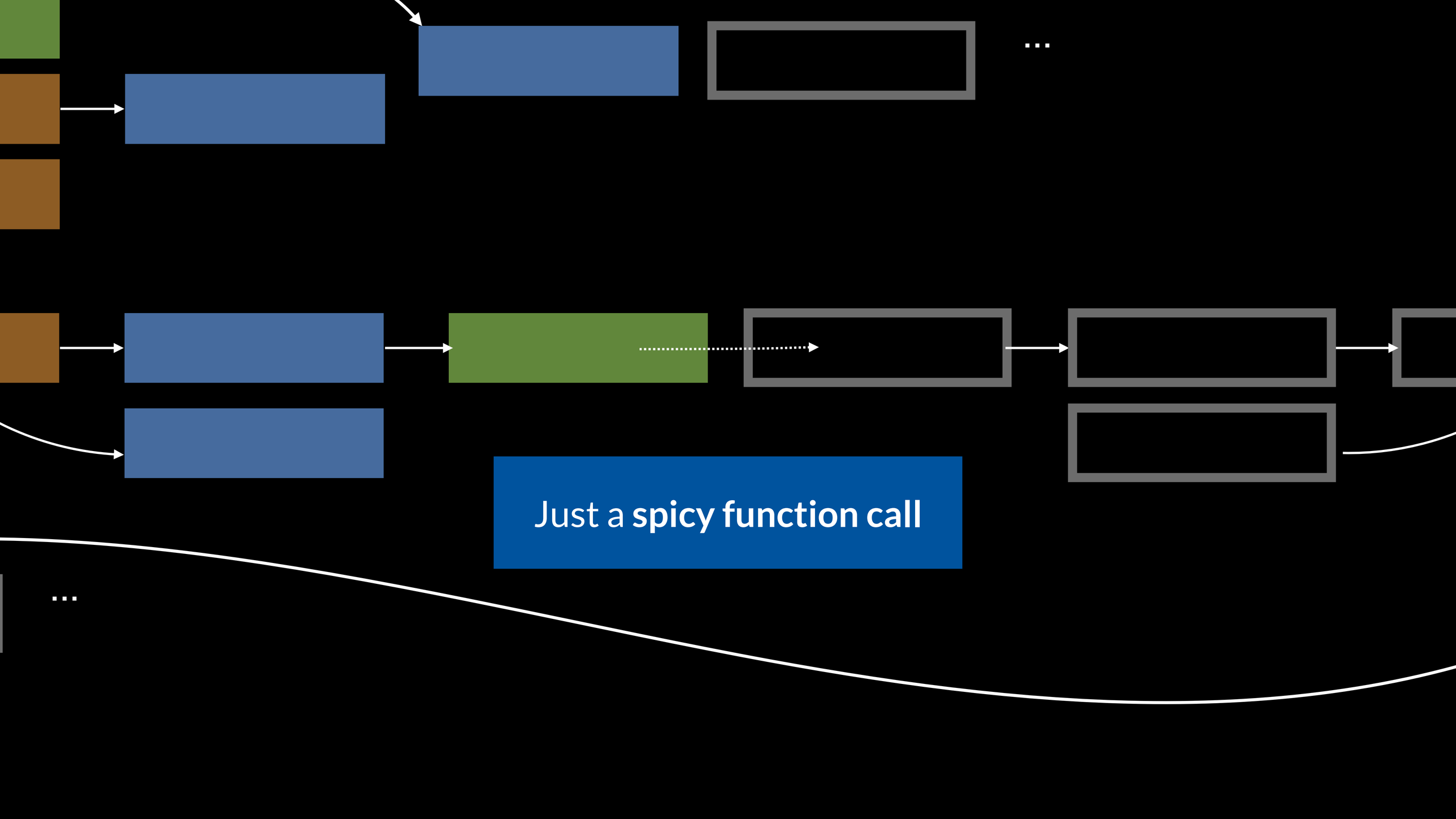
Live effect system

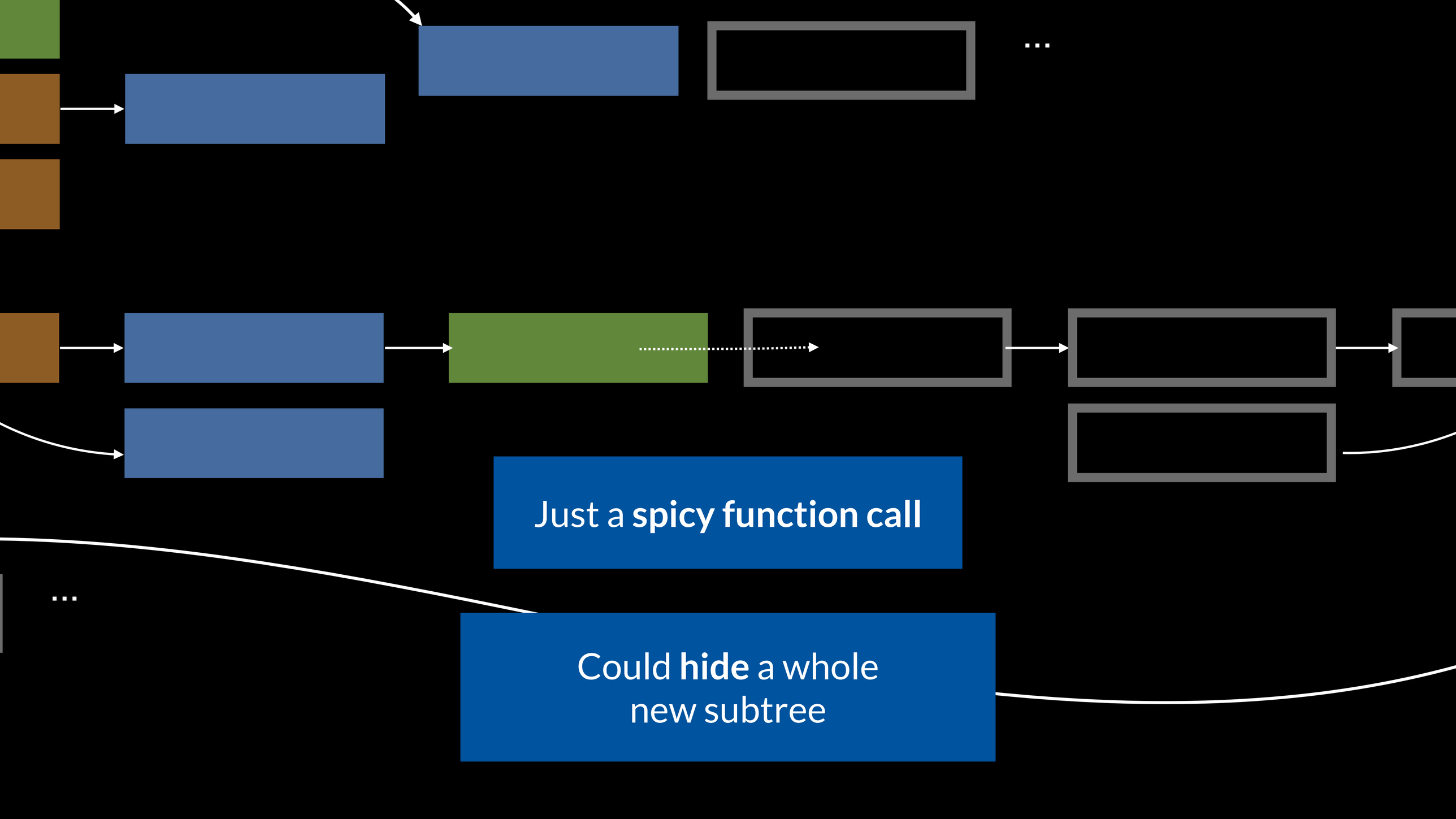


Live effect system

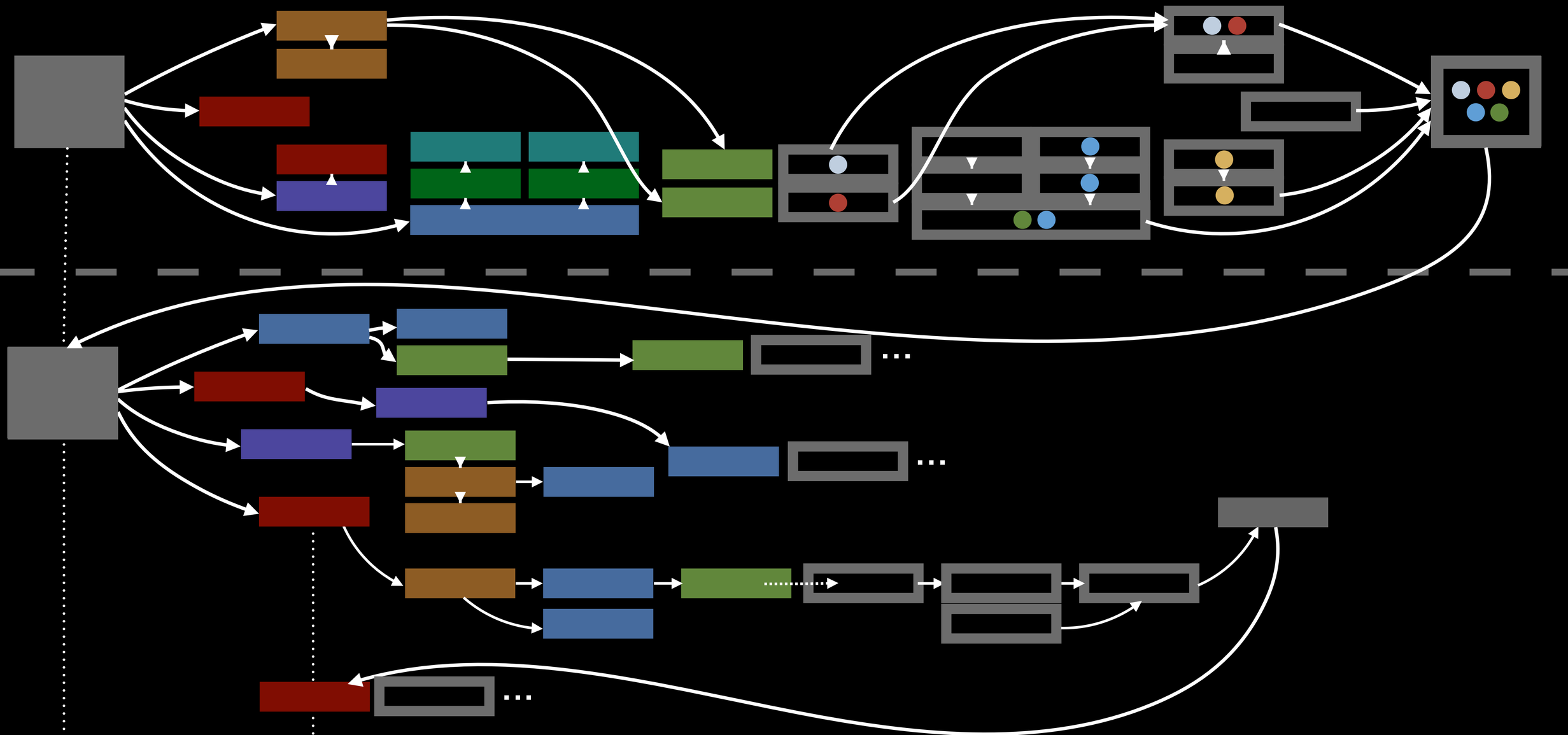




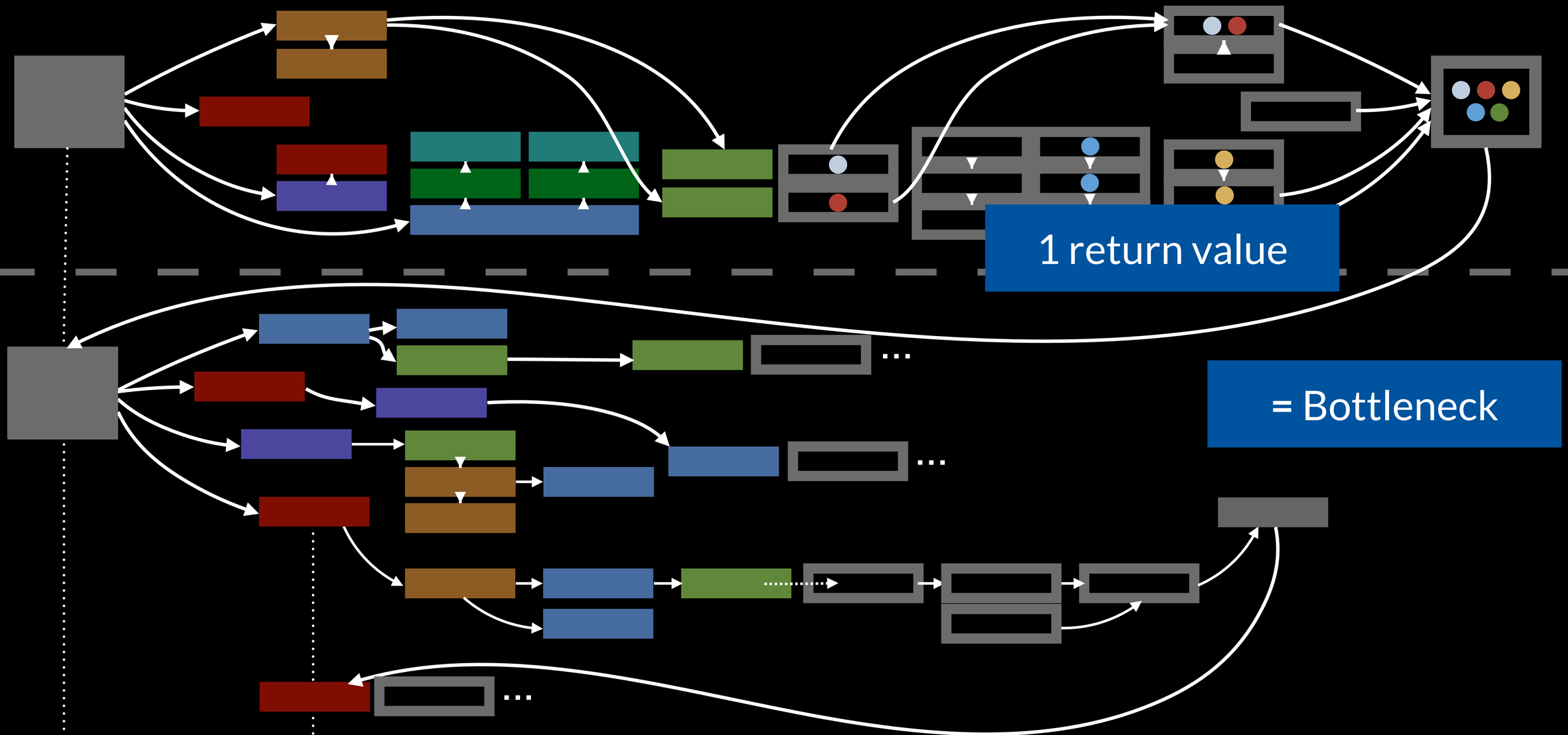

...
Just a **spicy function call**
...

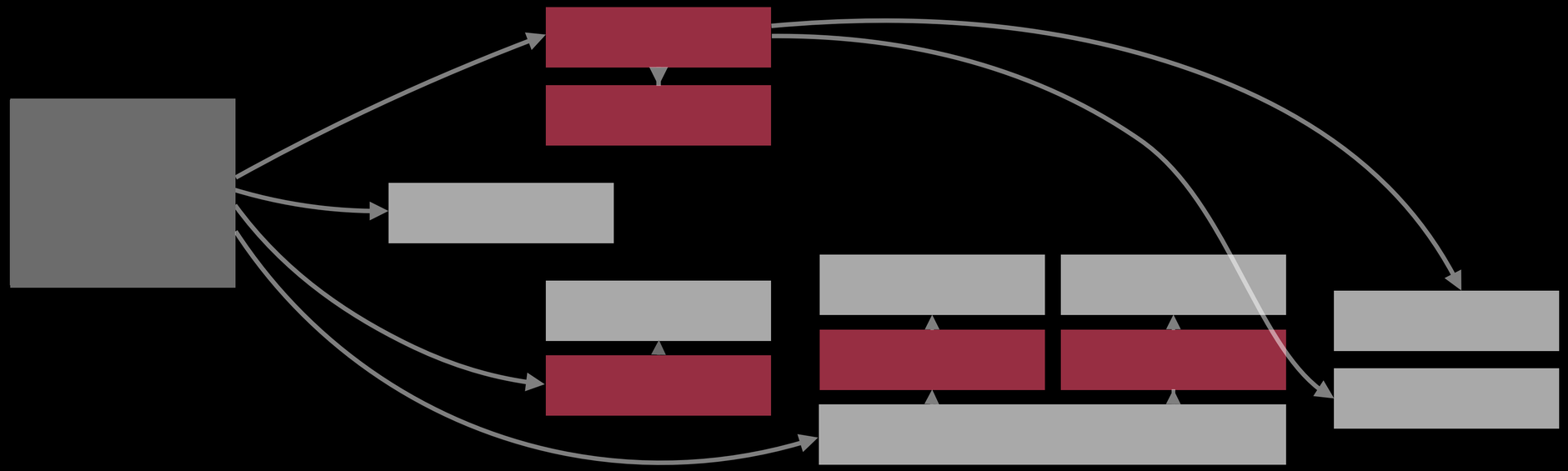

...
...
Just a **spicy function call**
Could **hide** a whole
new subtree

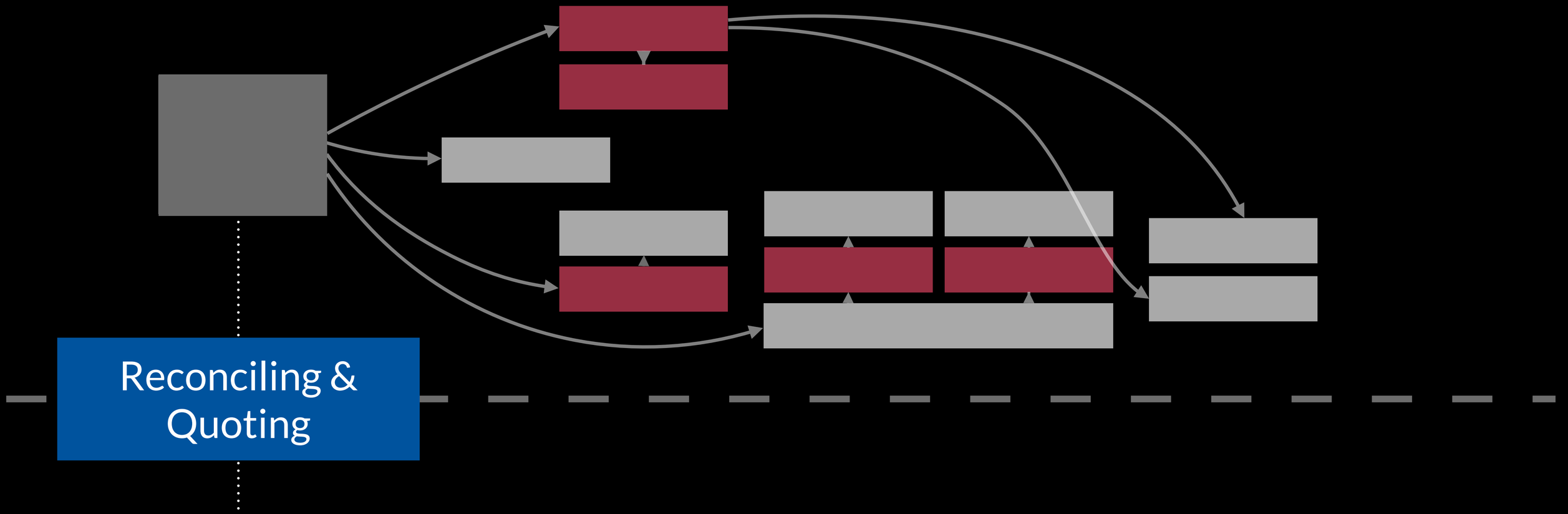
Live effect system

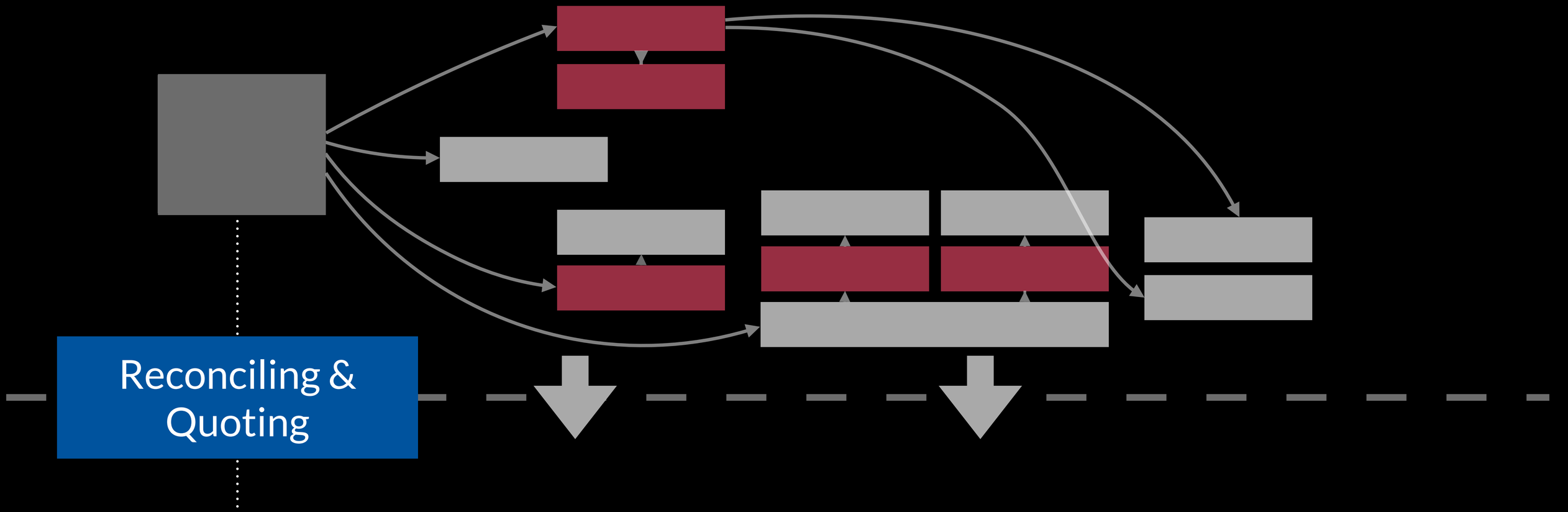


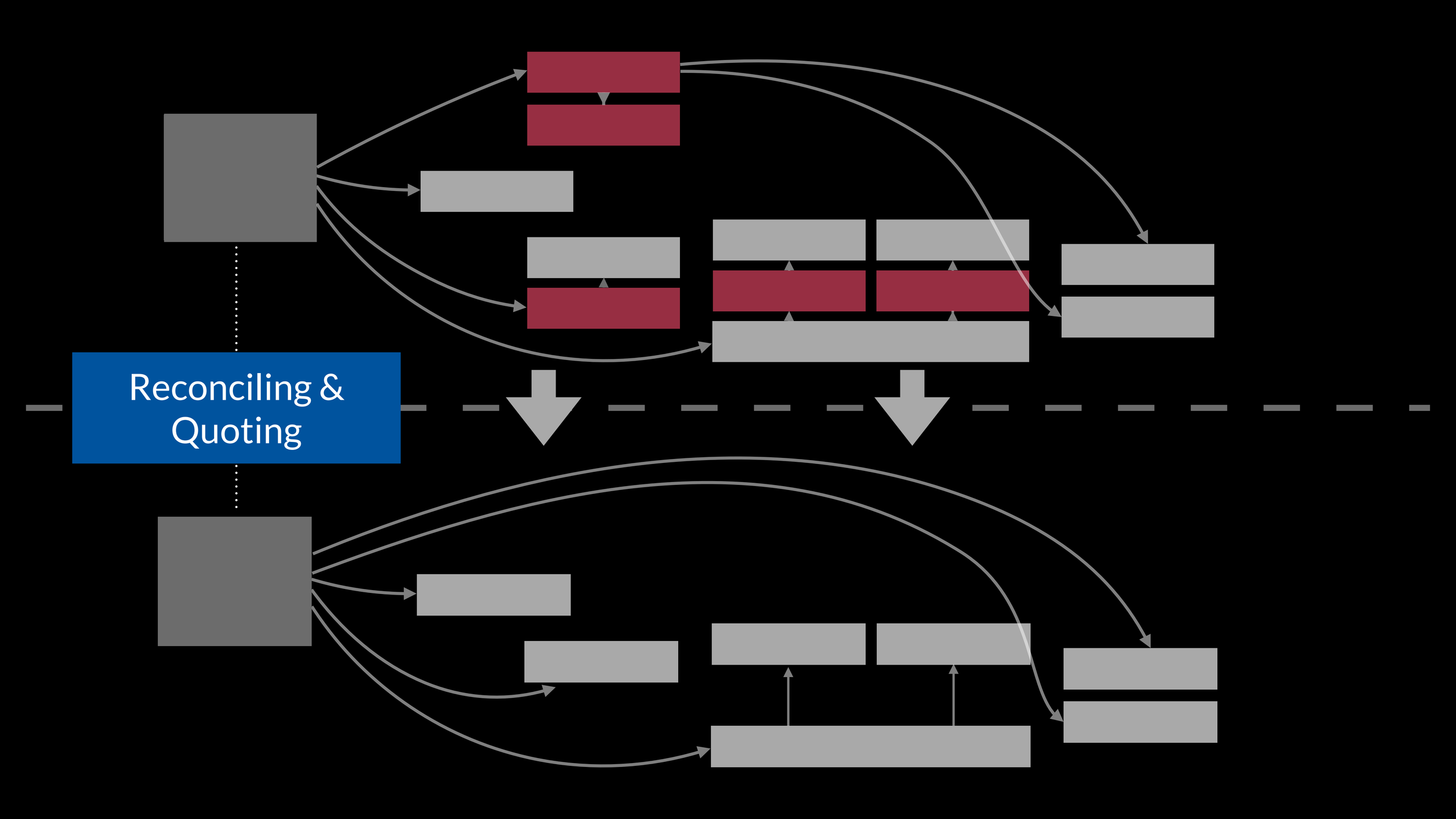
Live effect system

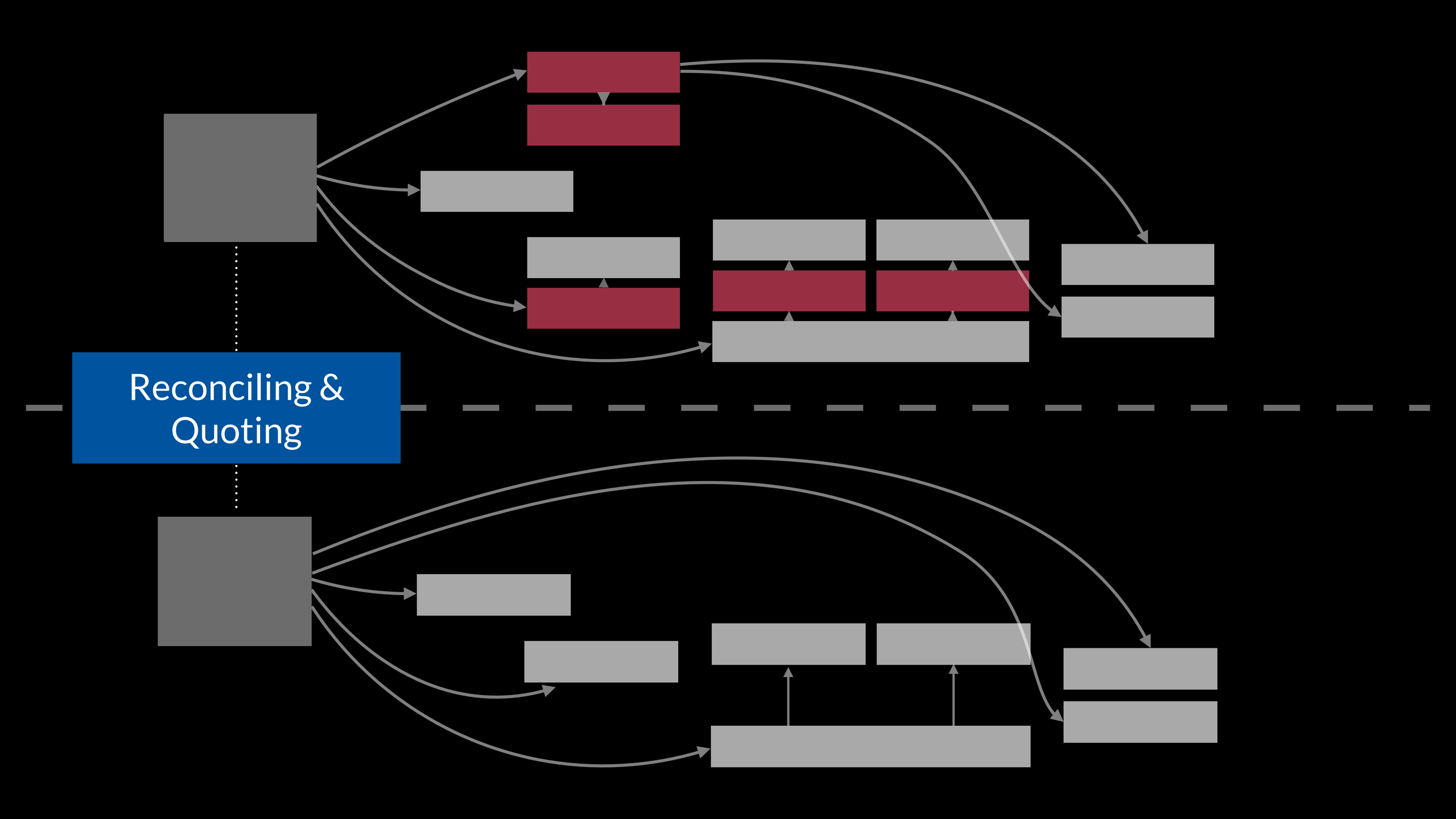


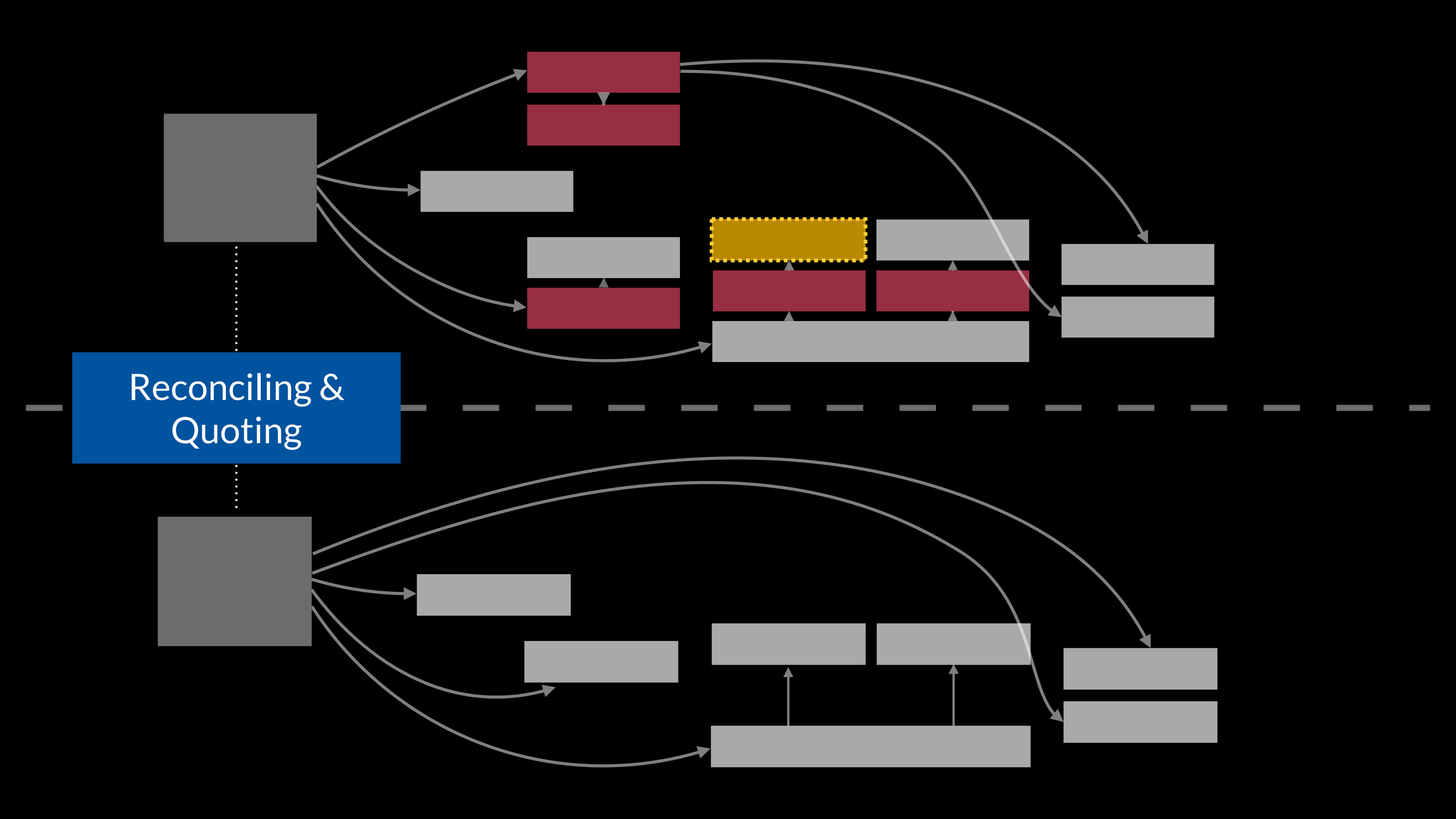


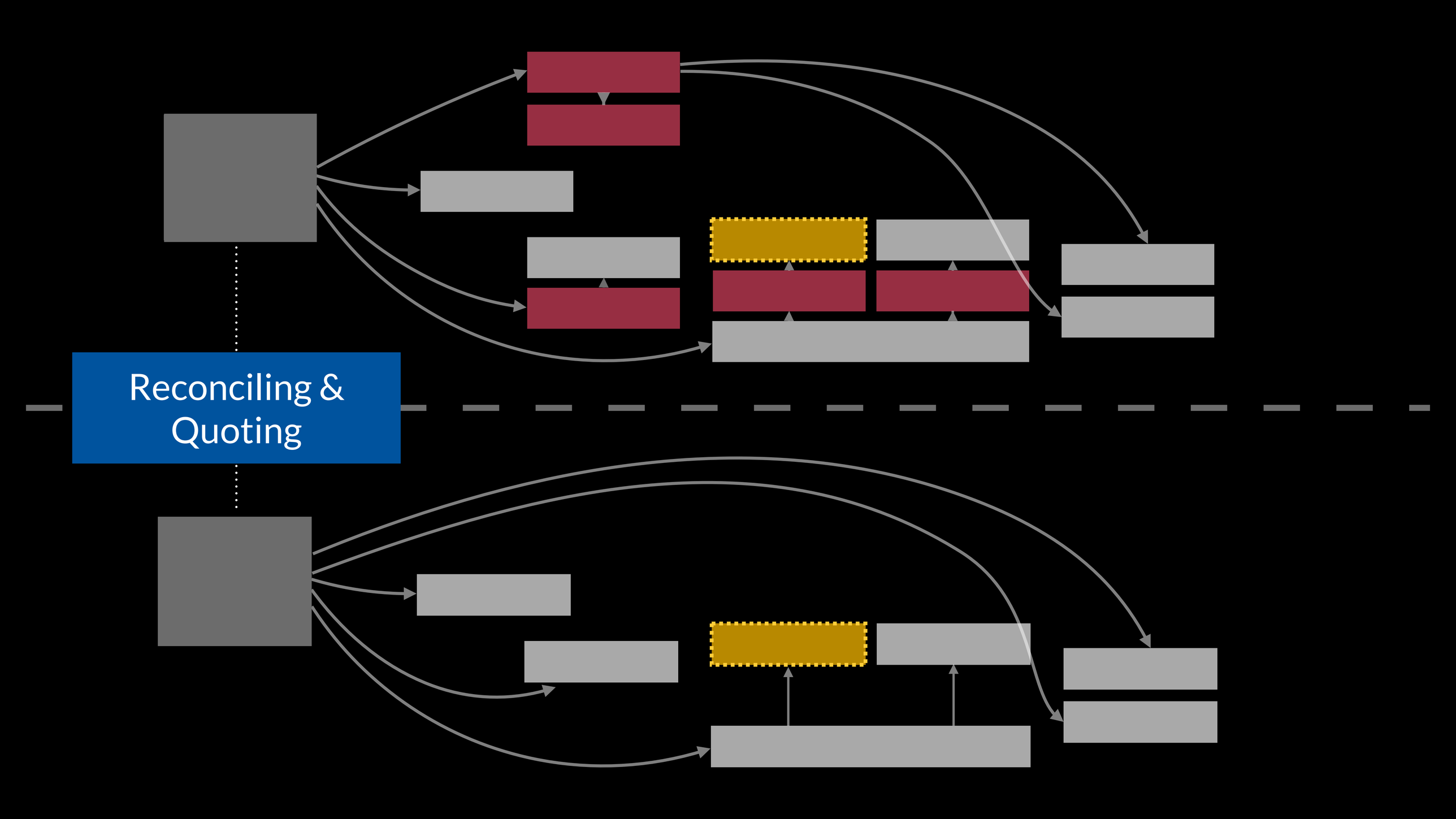

Reconciling &
Quoting

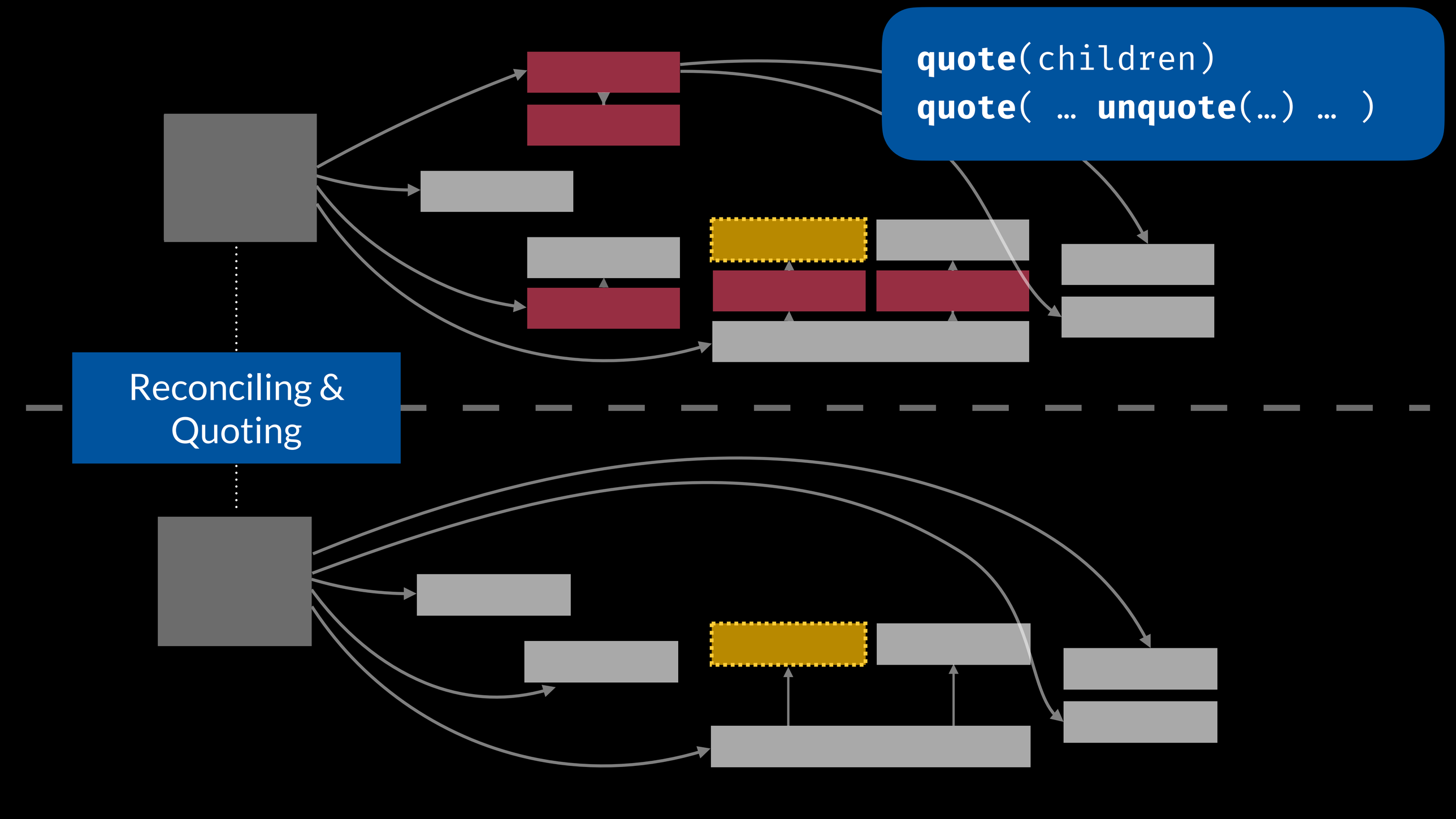

Reconciling &
Quoting

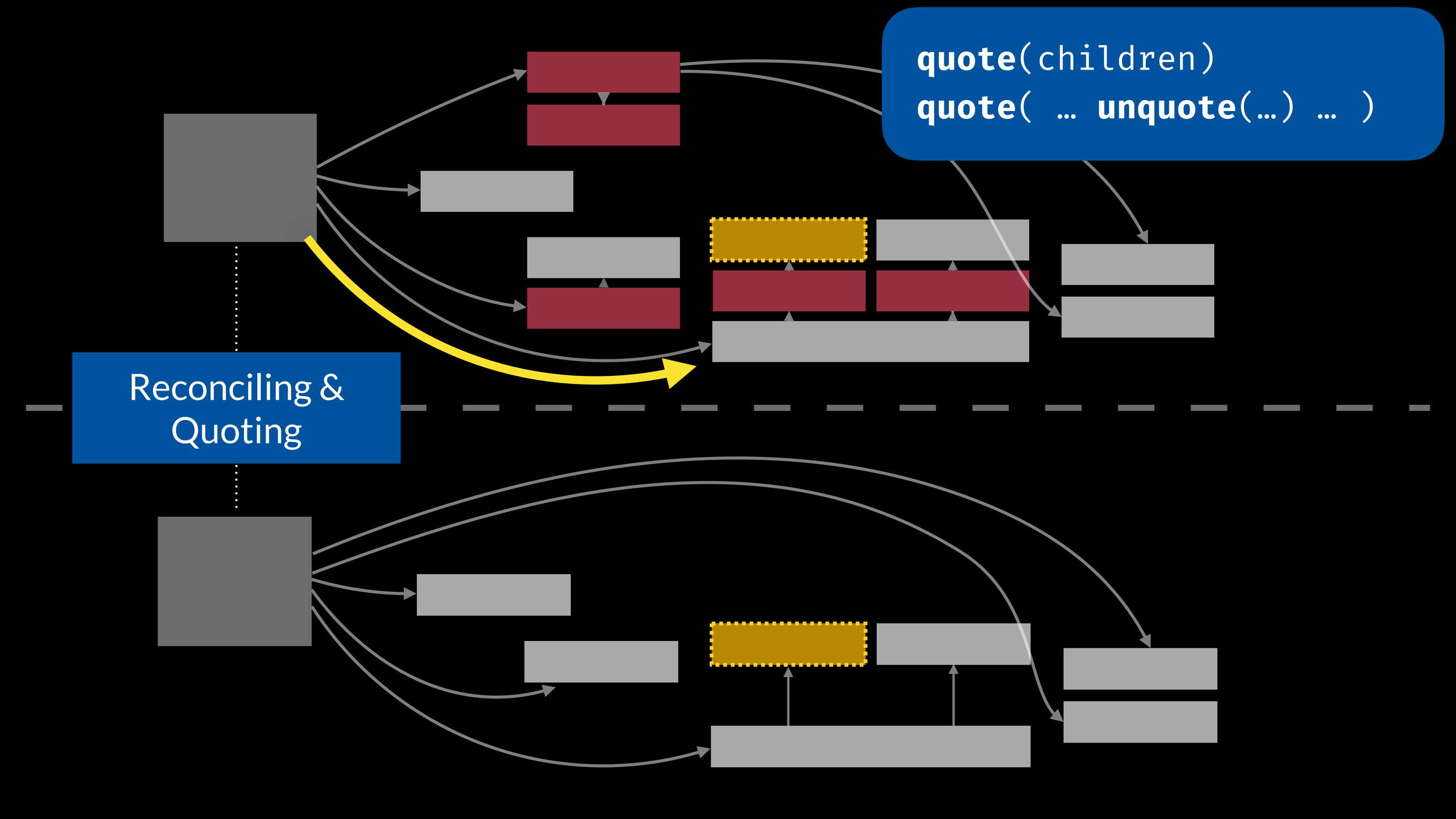

Reconciling &
Quoting

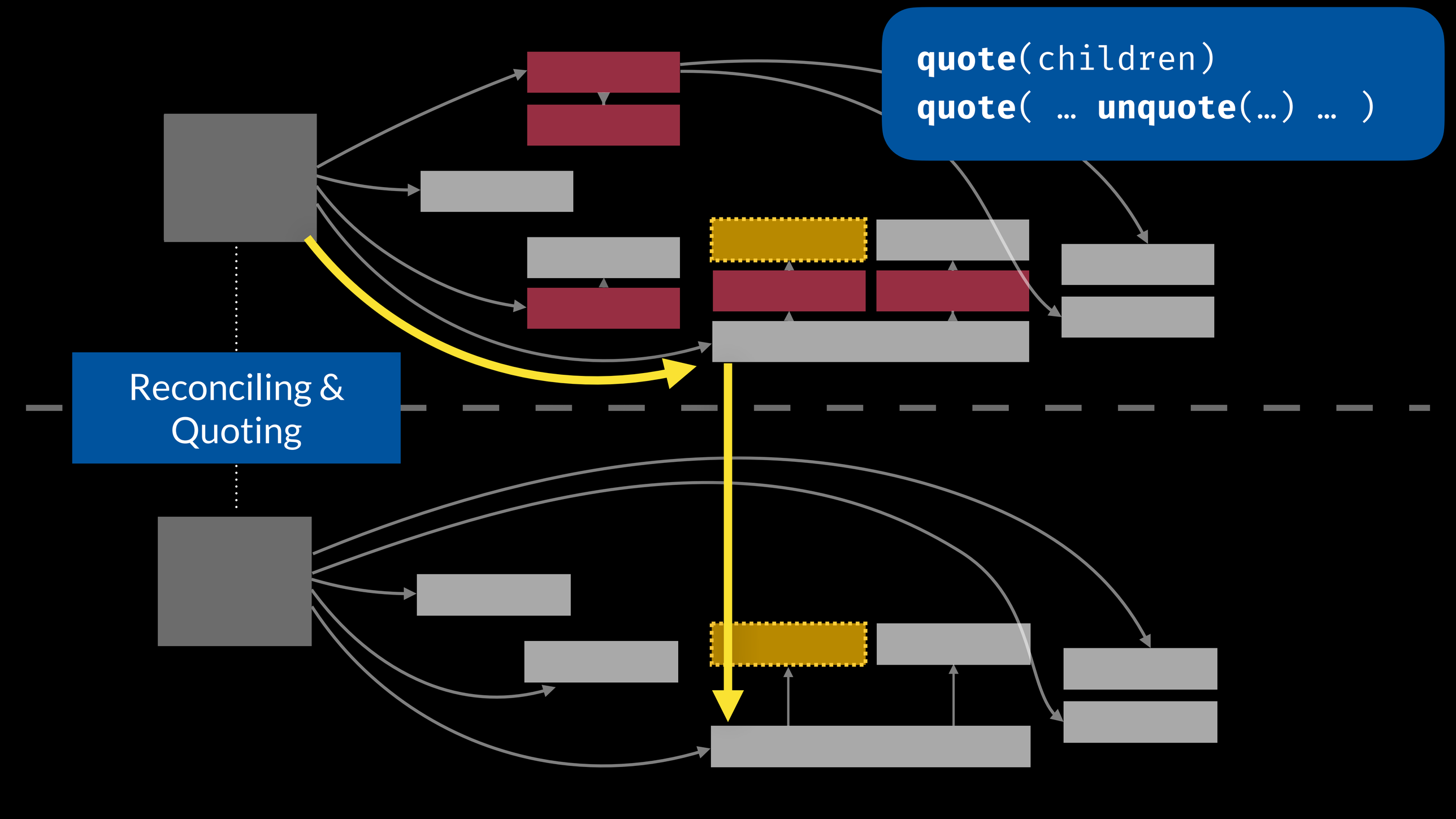

Reconciling &
Quoting

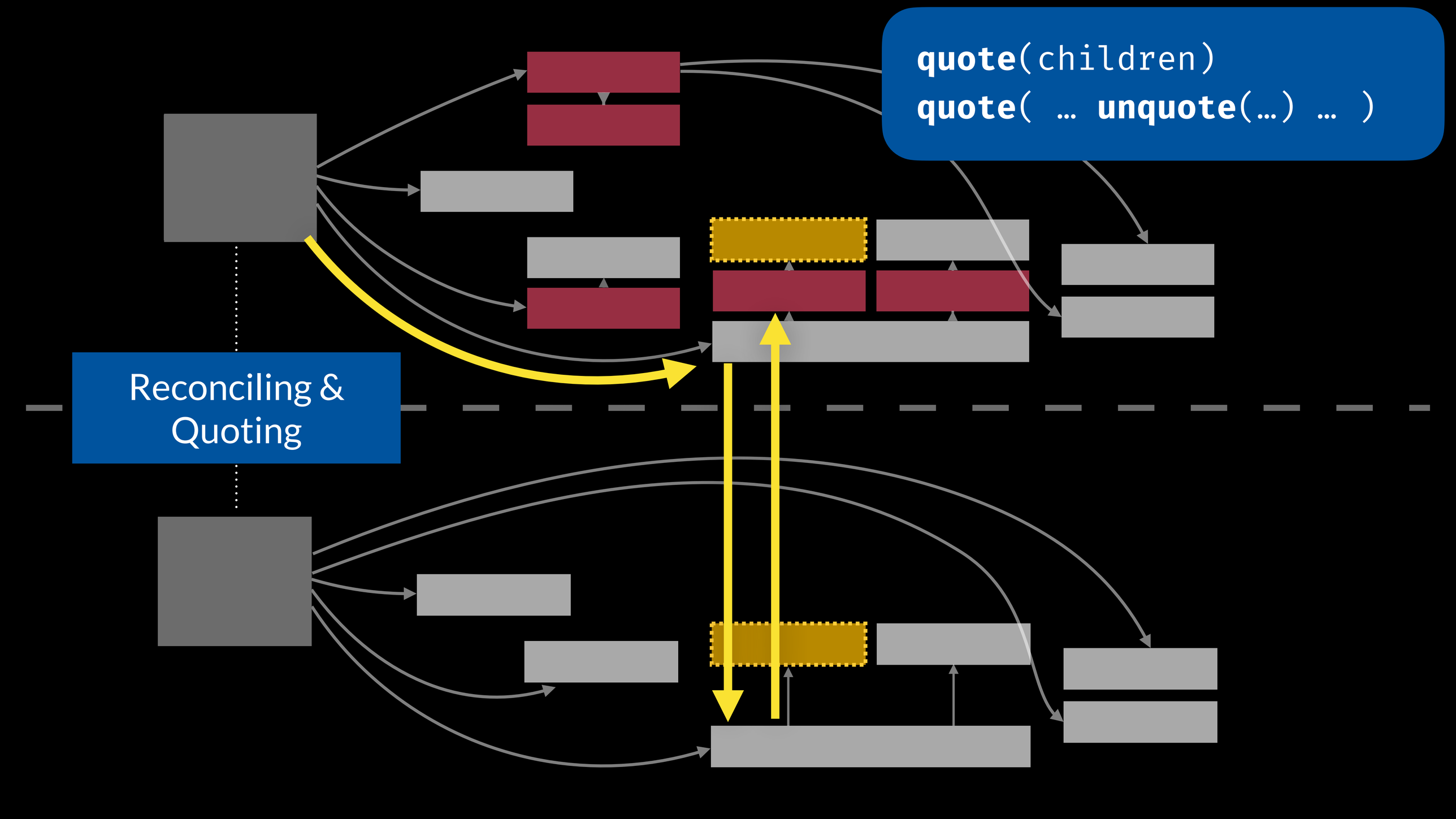

Reconciling &
Quoting

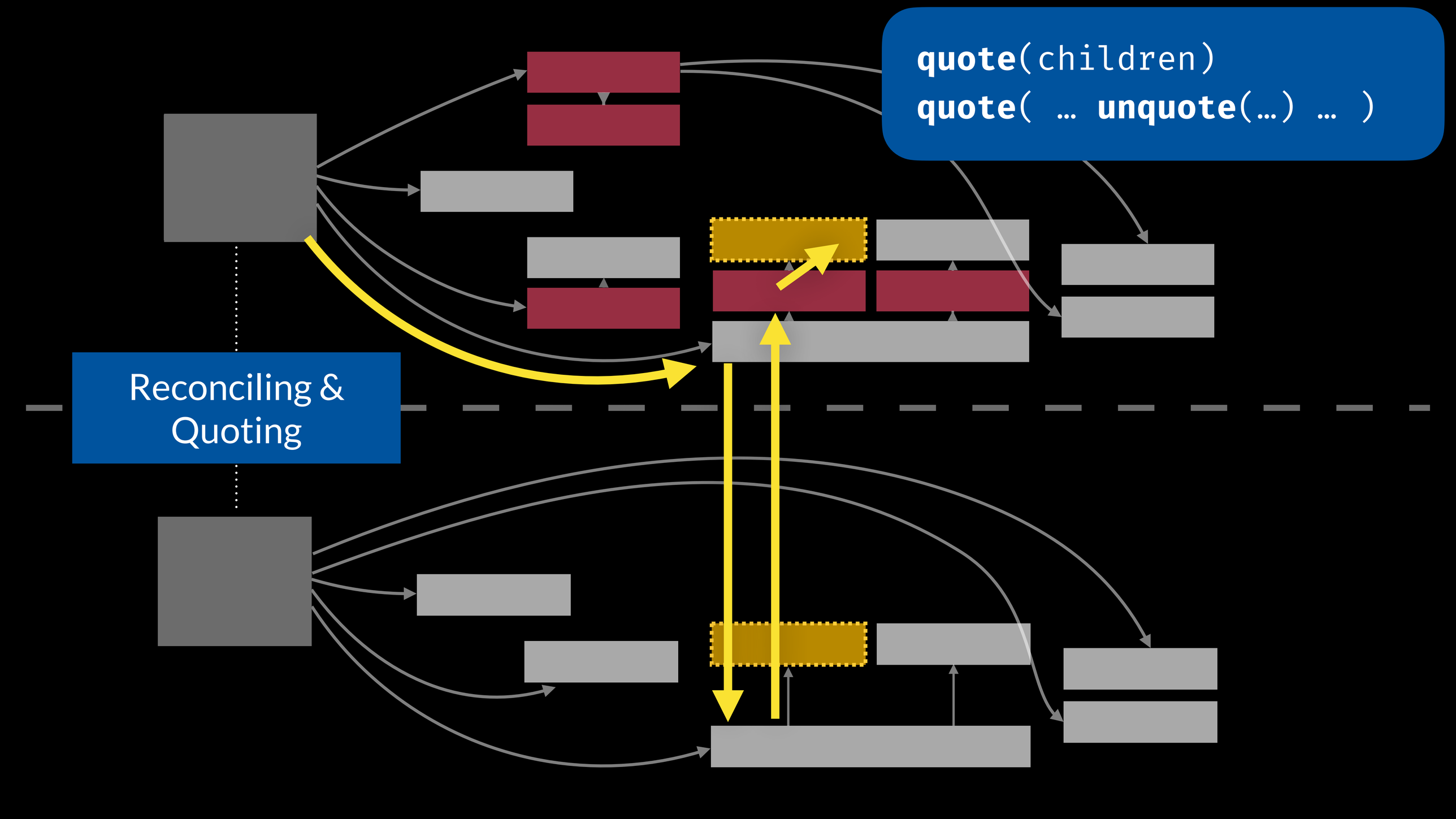

Reconciling &
Quoting

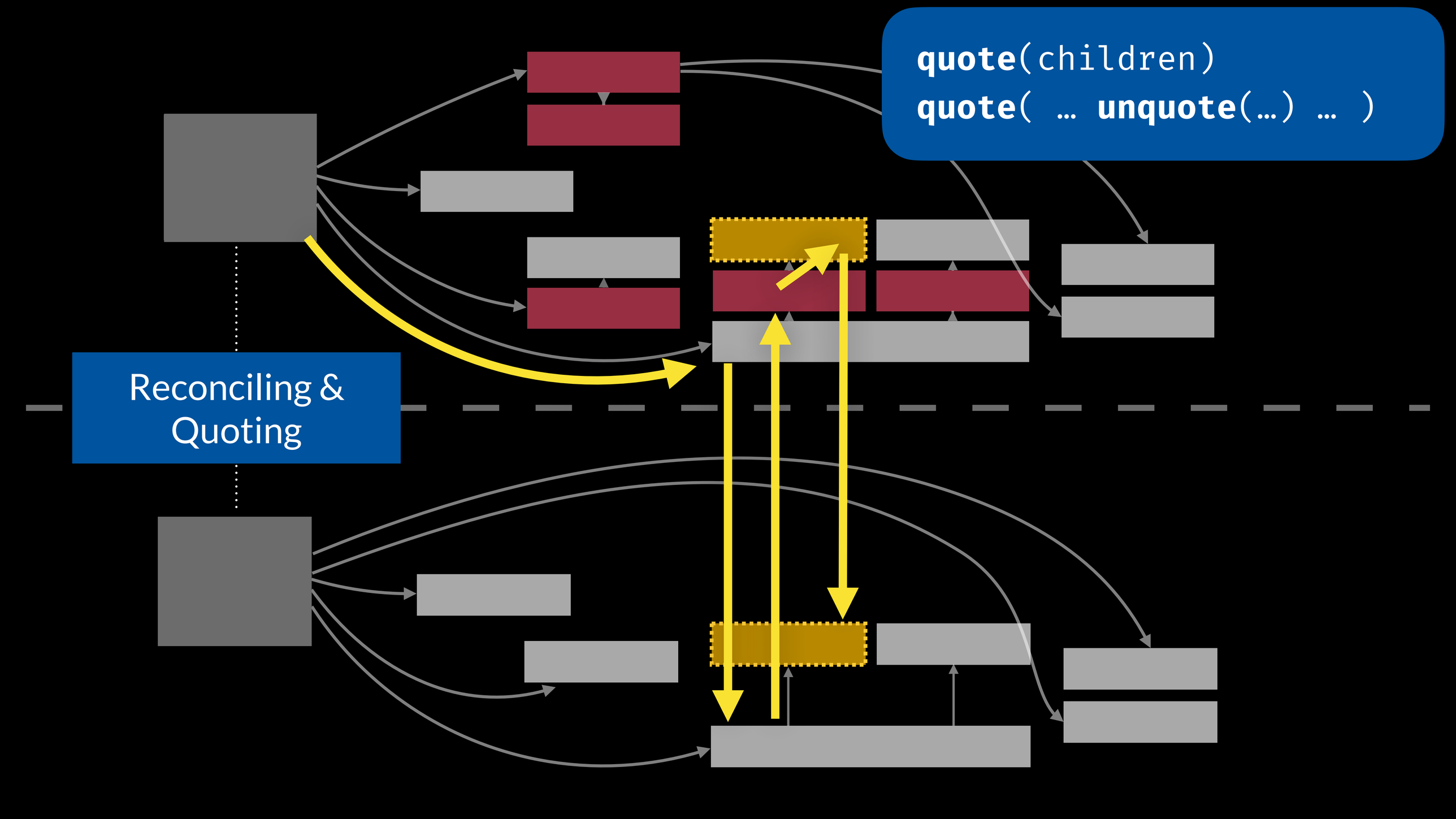

quote(children)
quote(… unquote(…) …)
Reconciling &
Quoting

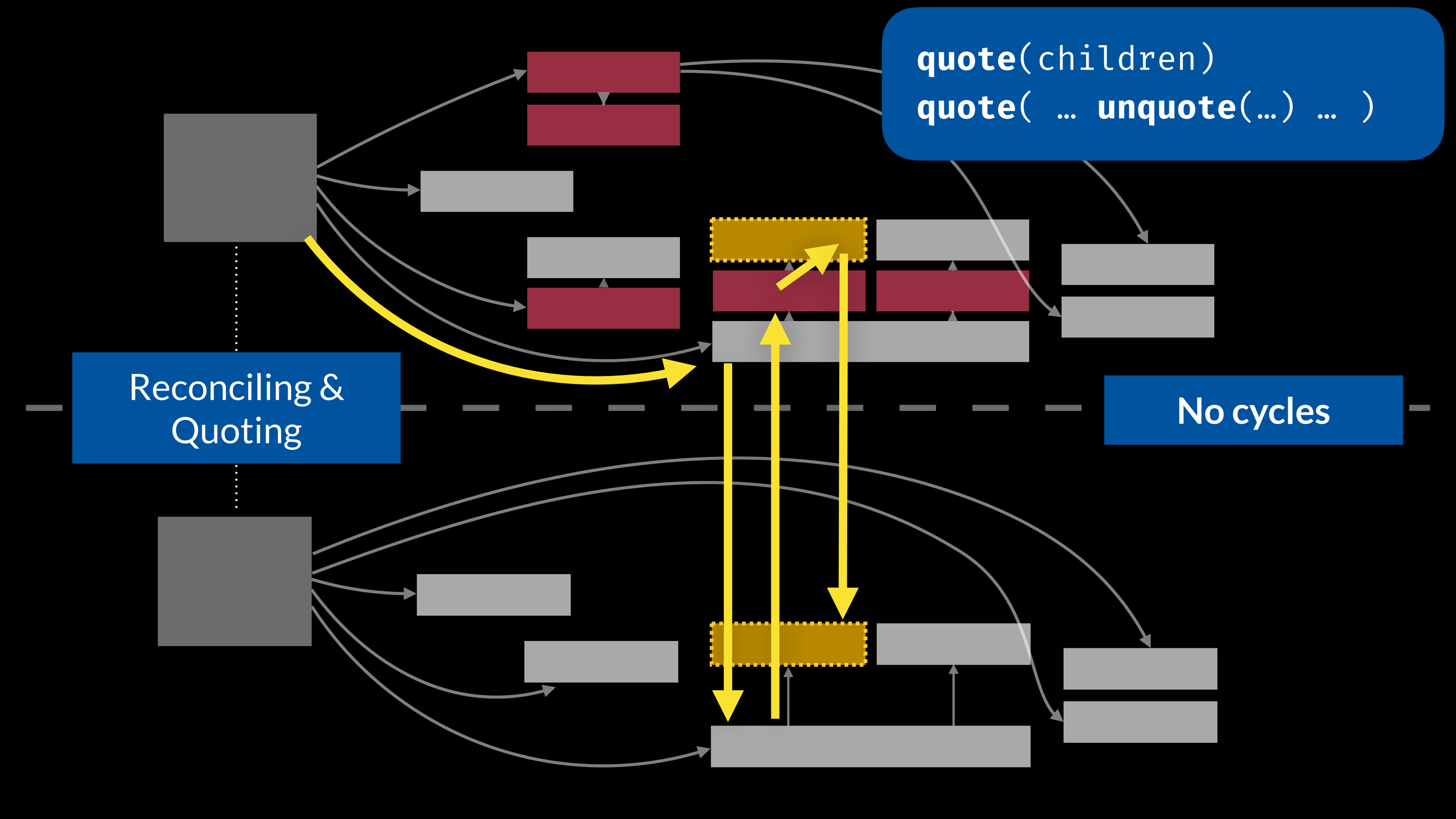

quote(children)
quote(… unquote(…) …)
Reconciling &
Quoting

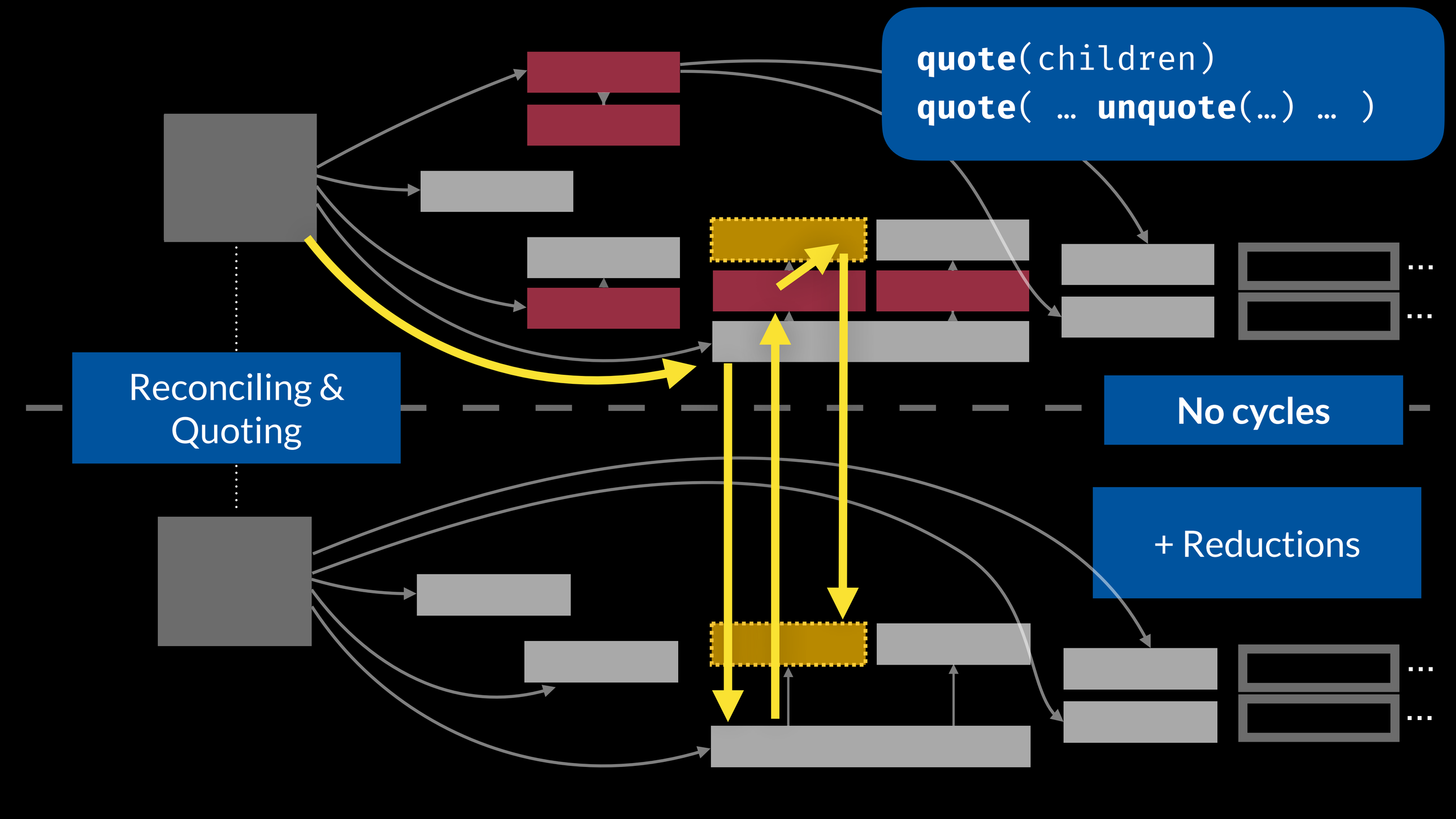

quote(children)
quote(… unquote(…) …)
Reconciling &
Quoting

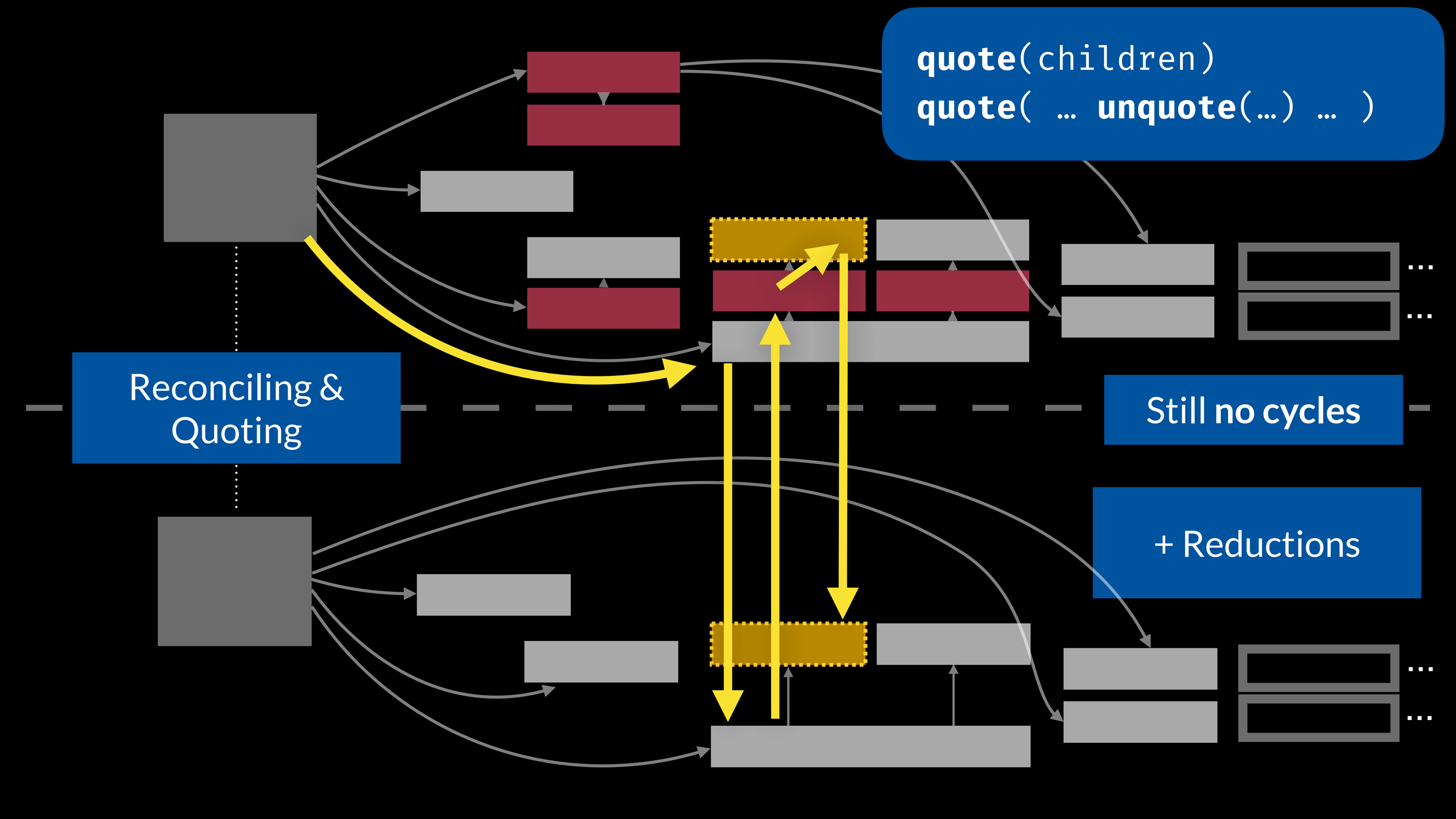

quote(children)
quote(… unquote(…) …)
Reconciling &
Quoting

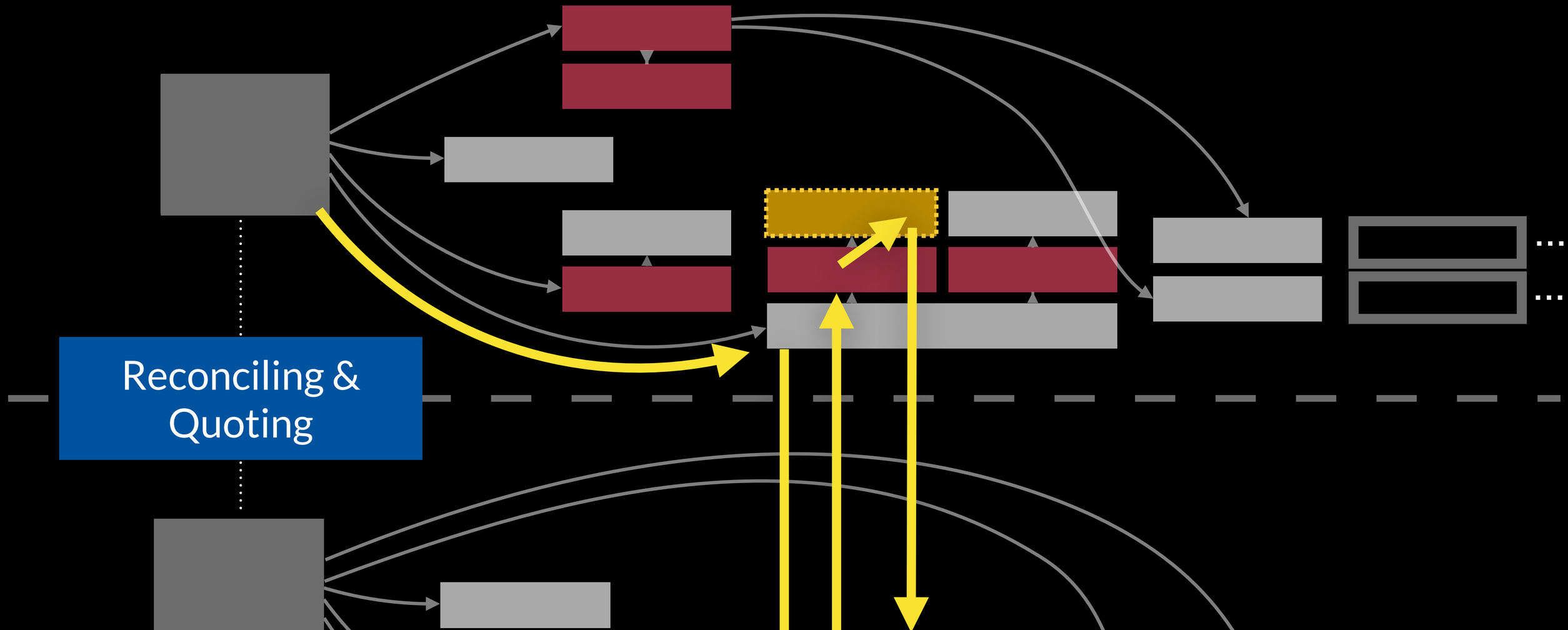

quote(children)
quote(… unquote(…) …)
Reconciling &
Quoting

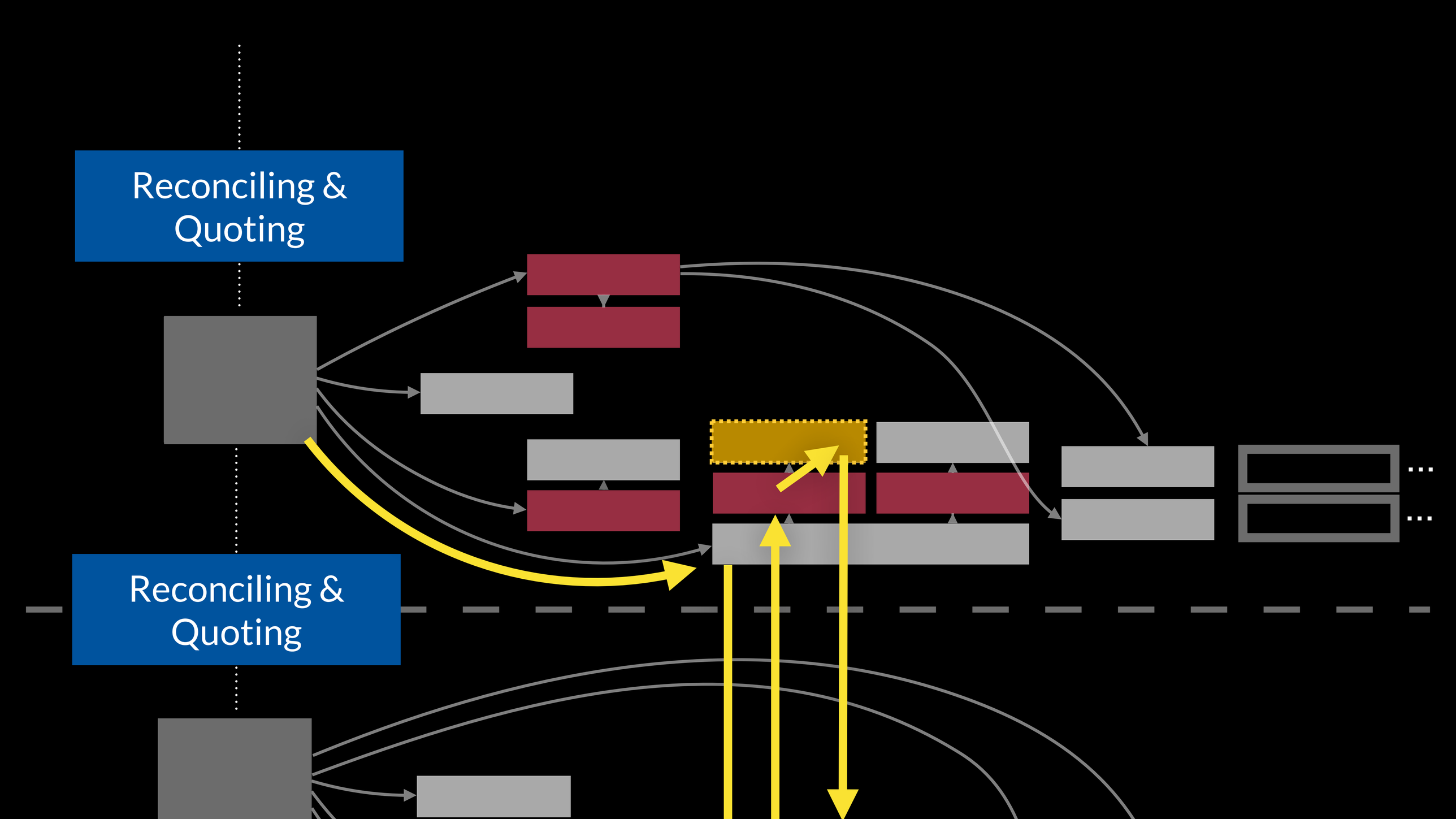

quote(children)
quote(… unquote(…) …)
Reconciling &
Quoting

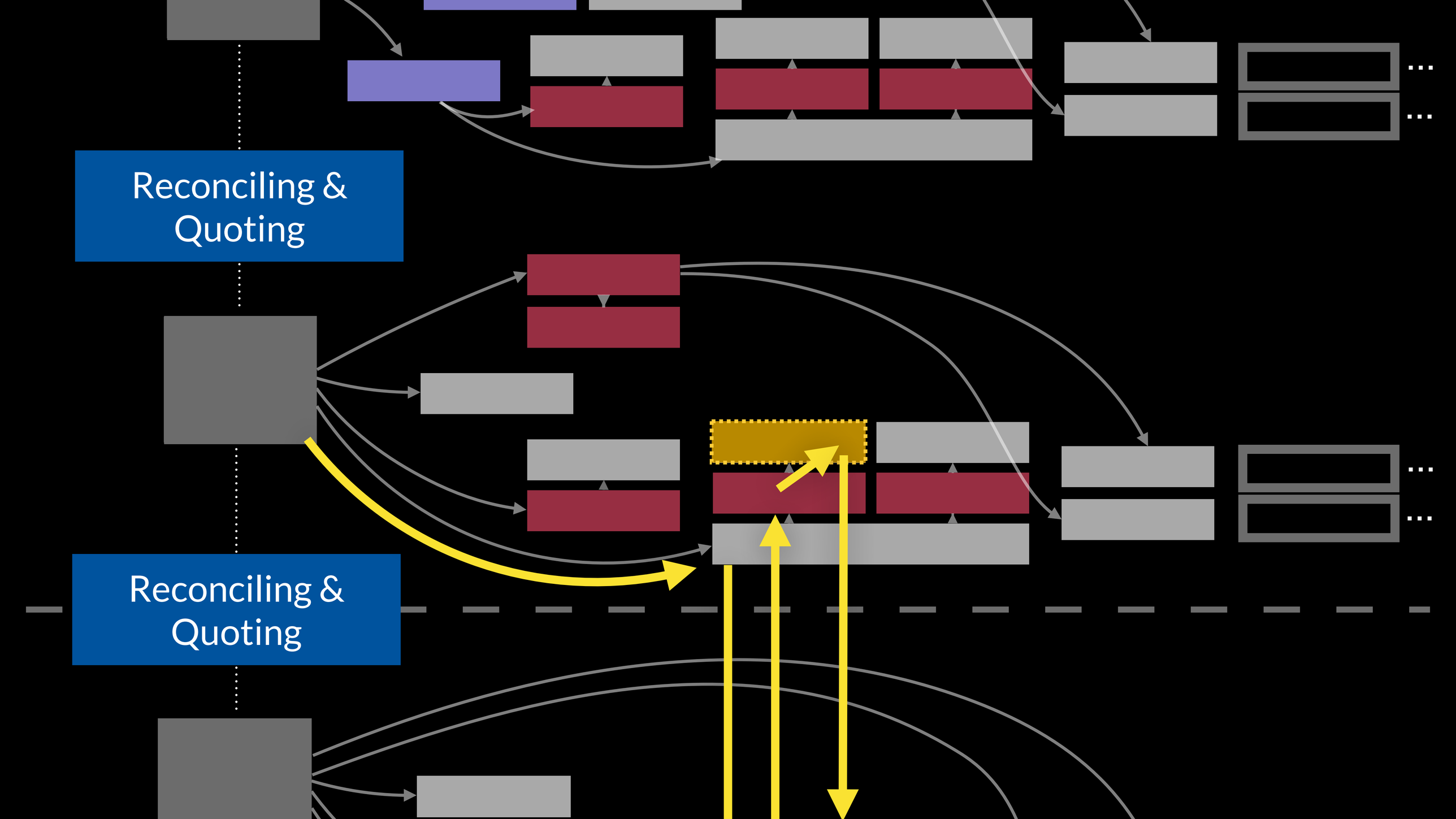

quote(children)
quote(… unquote(…) …)
Reconciling & Quoting
No cycles

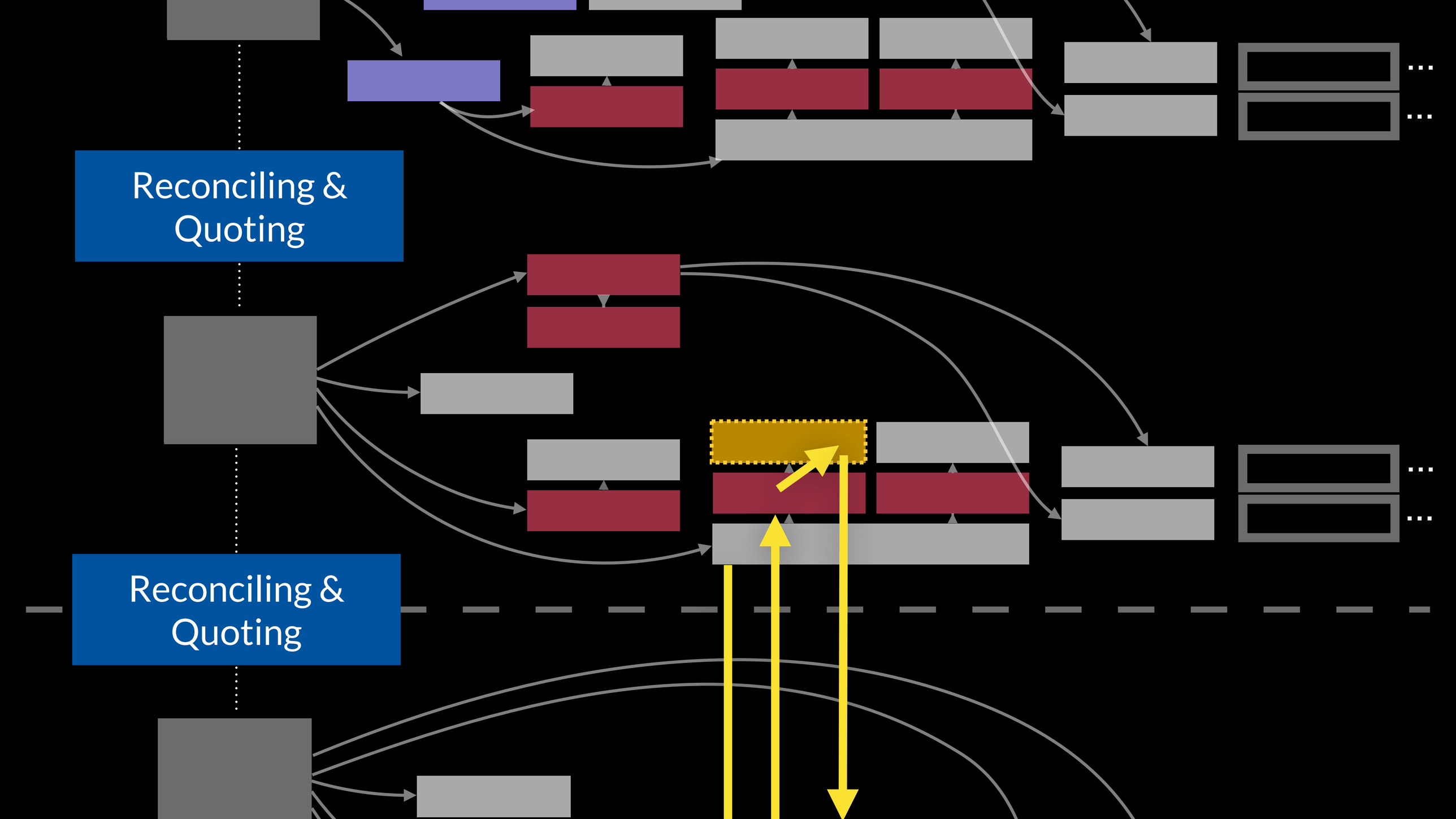

quote(children)
quote(… unquote(…) …)
Reconciling & Quoting
No cycles
+ Reductions

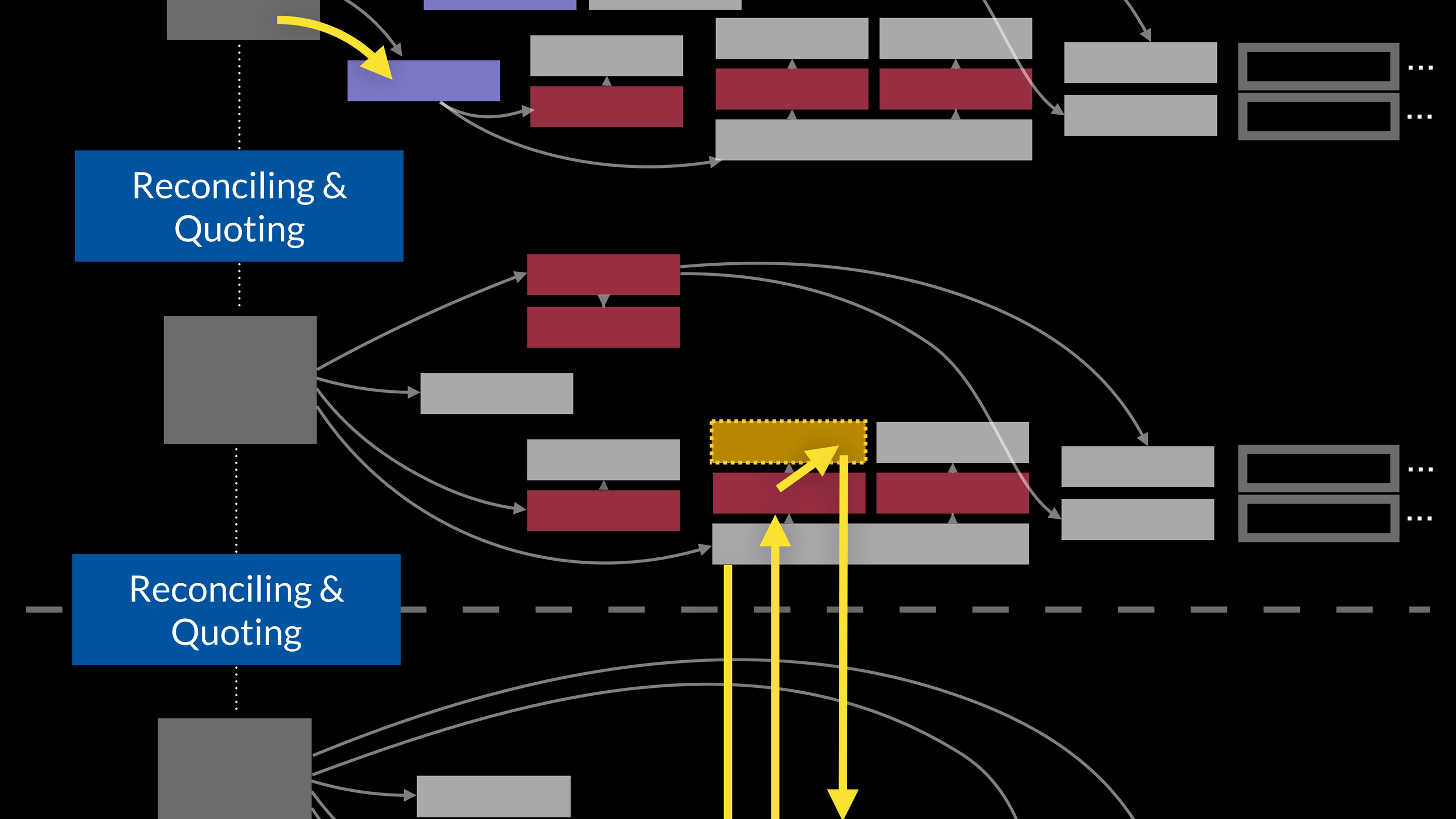

quote(children)
quote(… unquote(…) …)
Reconciling & Quoting
Still no cycles
+ Reductions

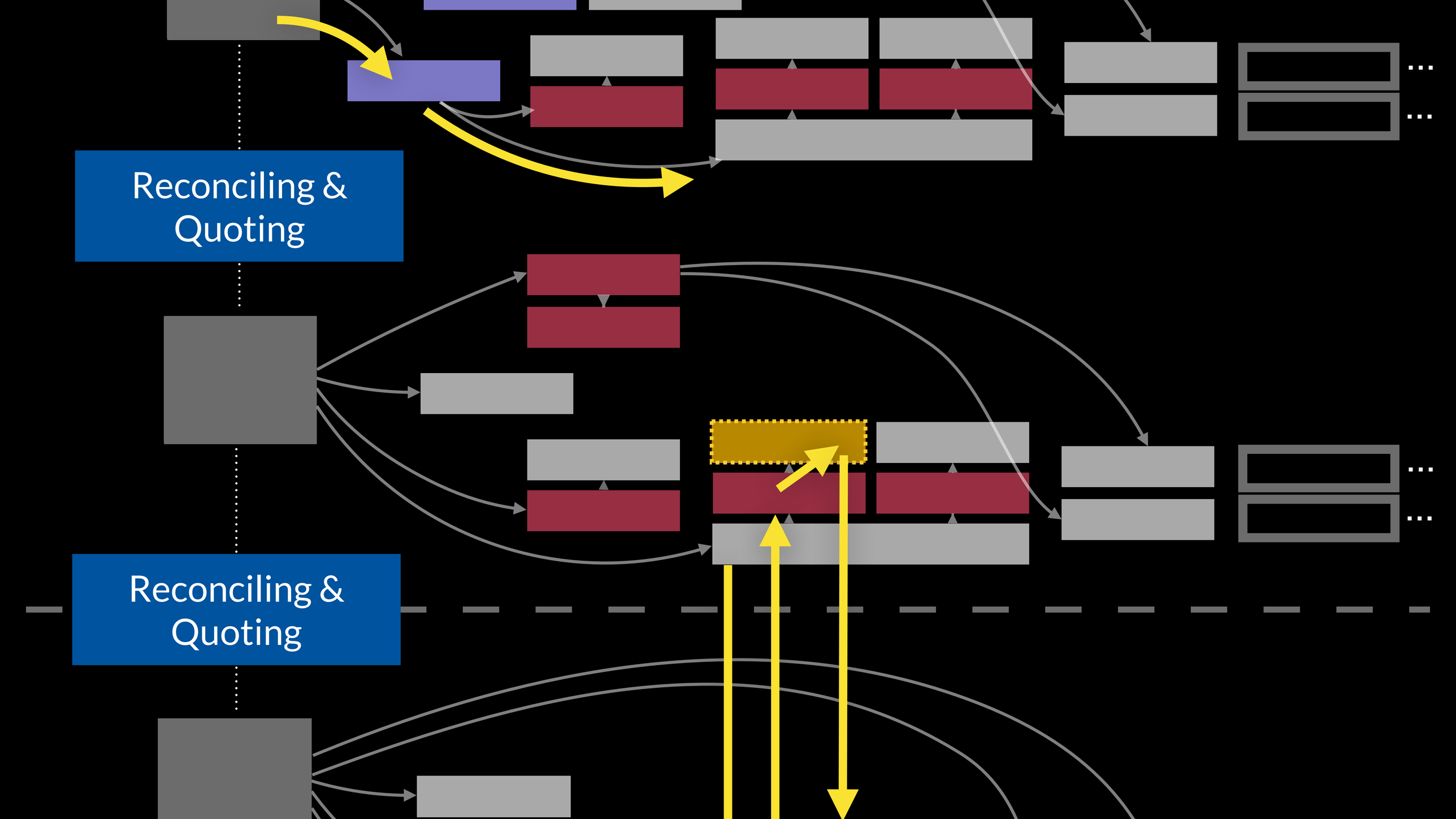

Reconciling &
Quoting

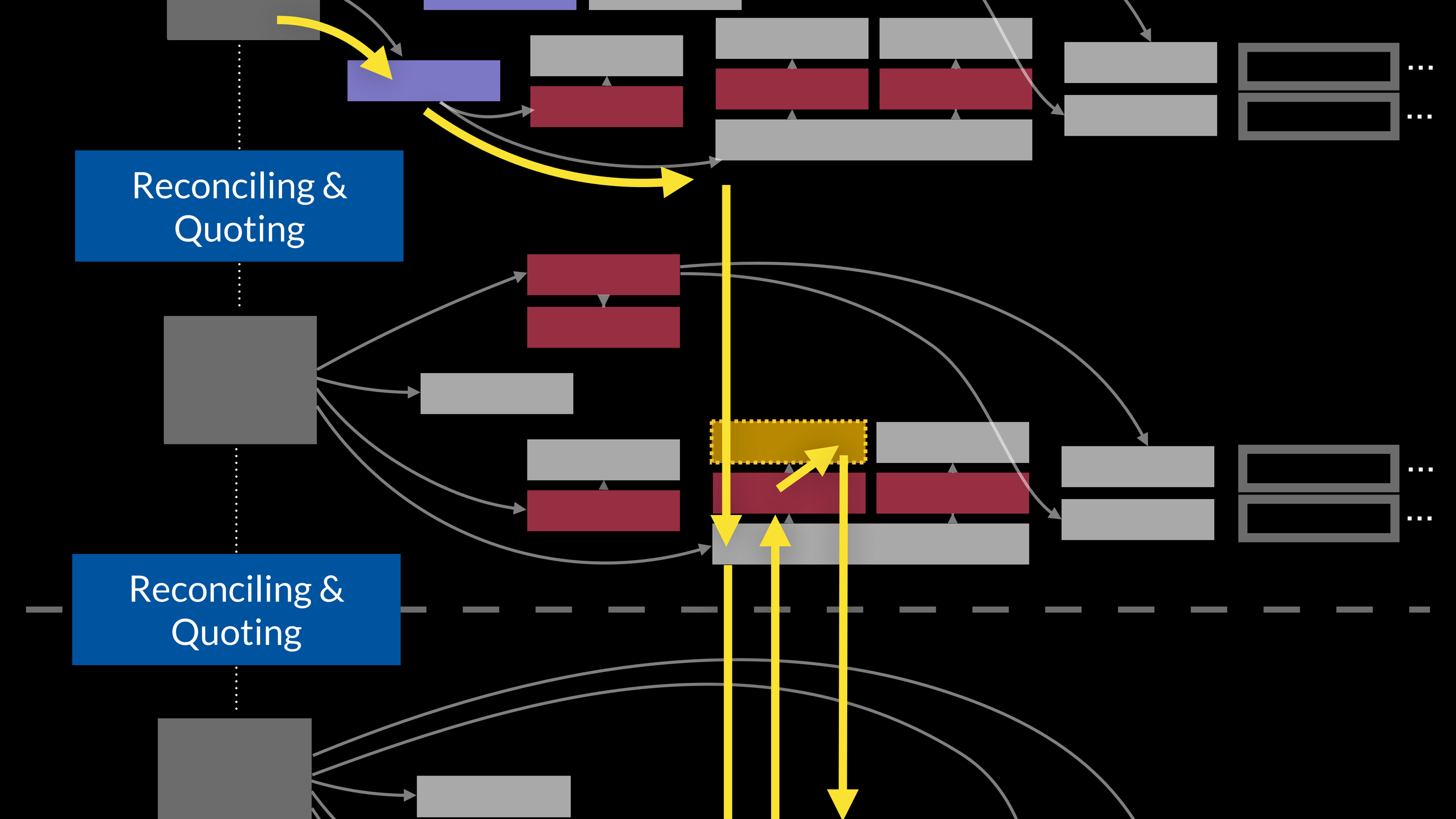

Reconciling &
Quoting
Reconciling &
Quoting

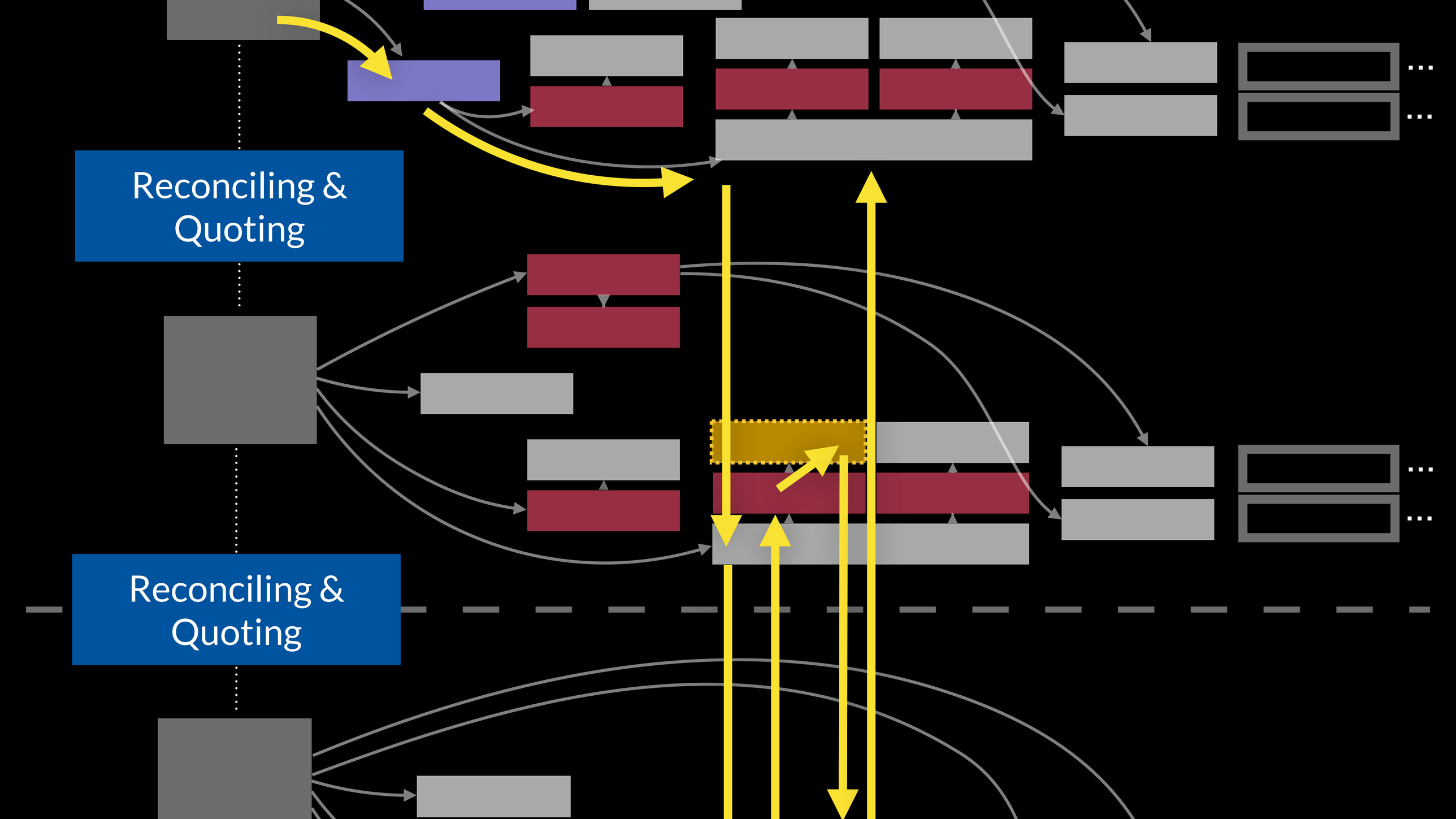

Reconciling &
Quoting
Reconciling &
Quoting

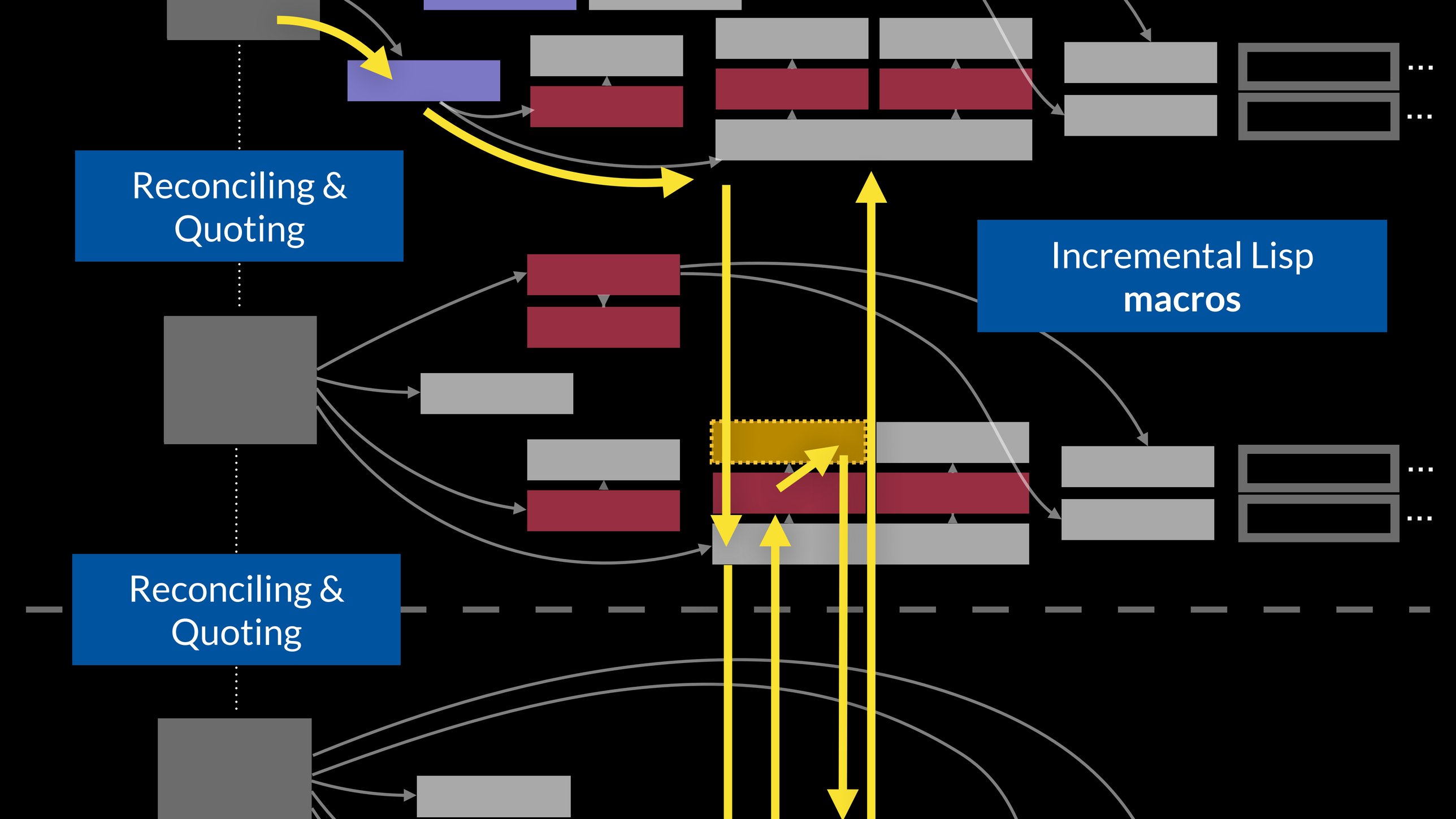

Reconciling &
Quoting
Reconciling &
Quoting

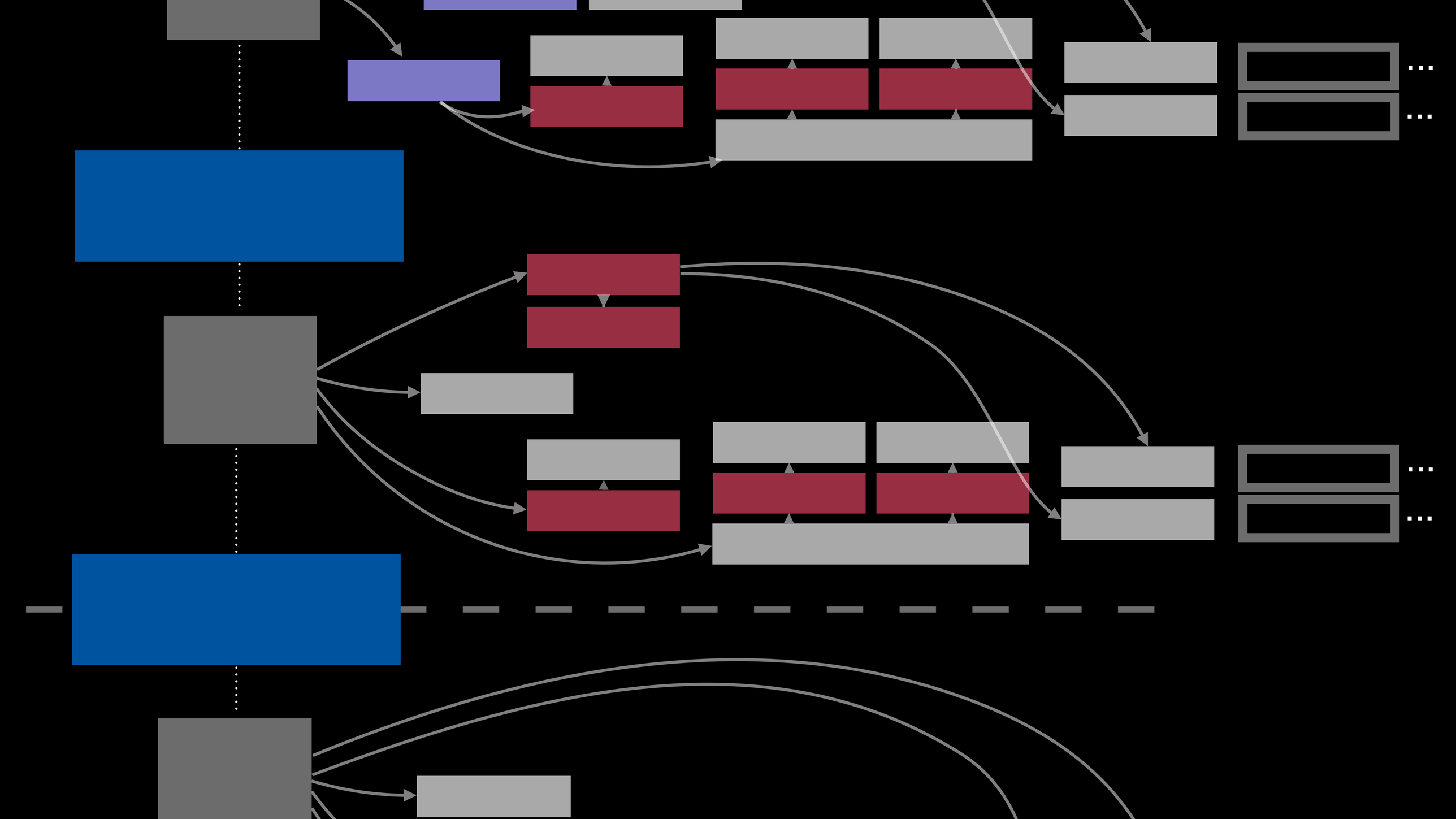

Reconciling &
Quoting
Reconciling &
Quoting

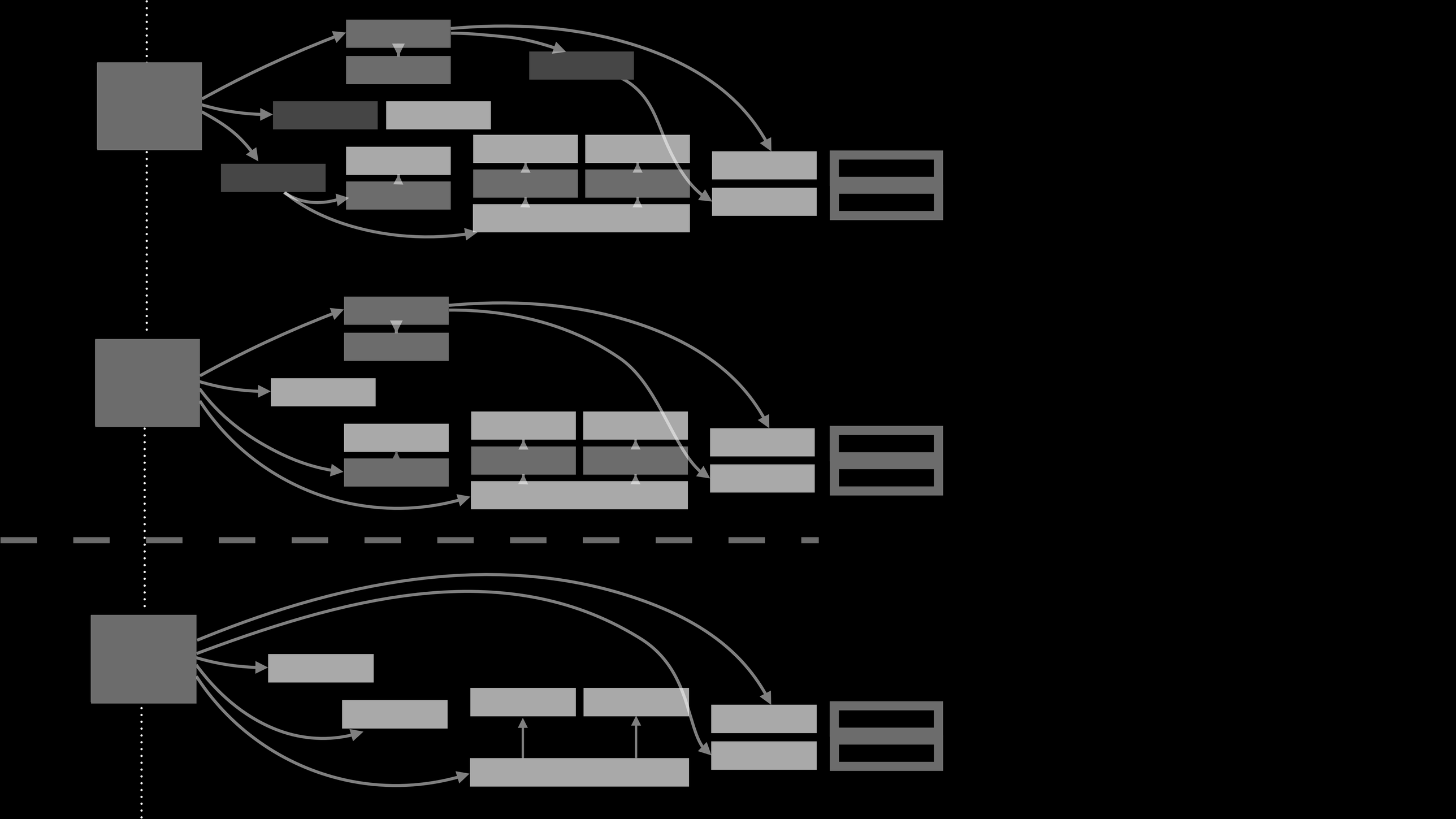

Reconciling &
Quoting
Reconciling &
Quoting

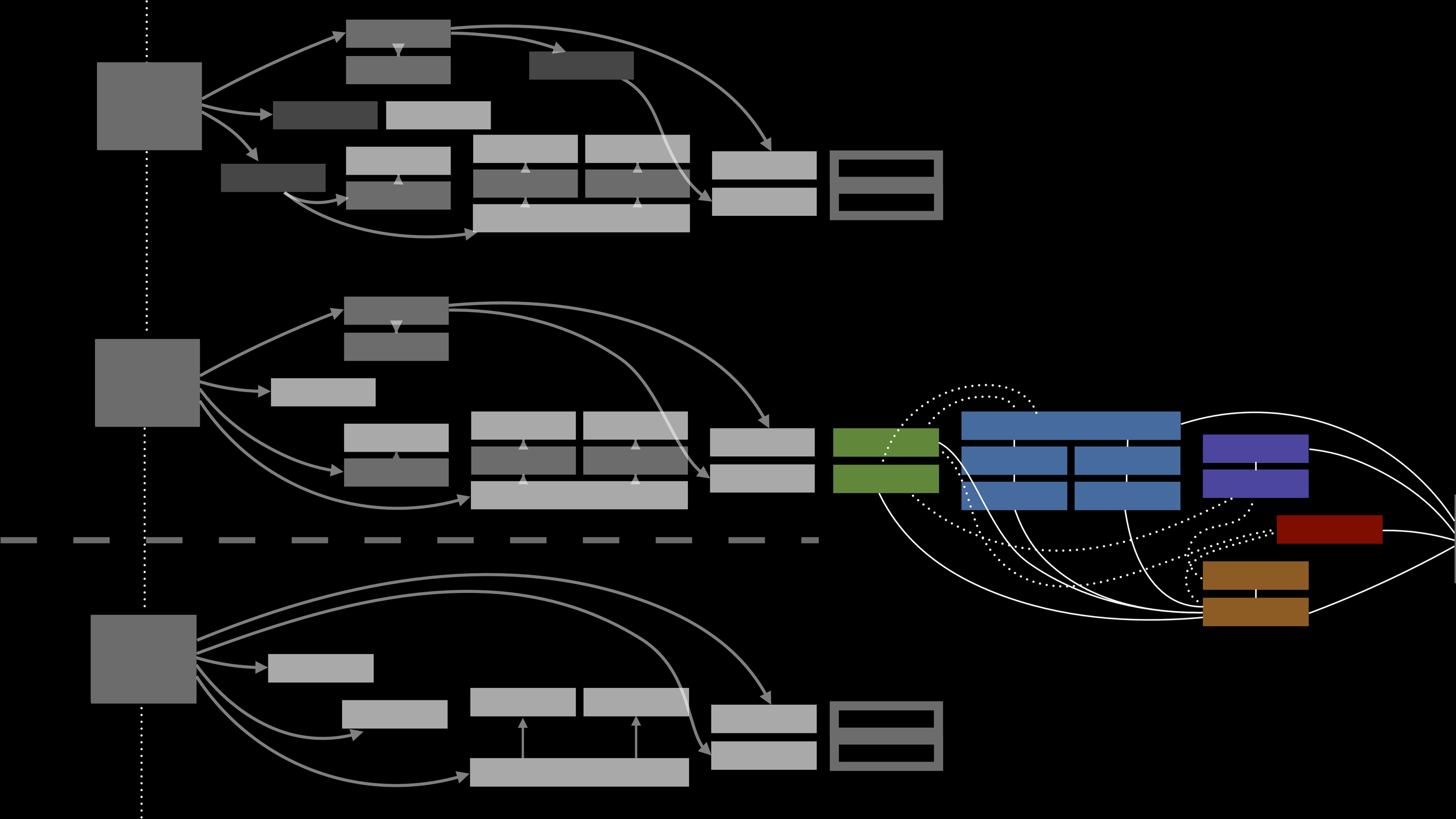

Reconciling &
Quoting
Reconciling &
Quoting

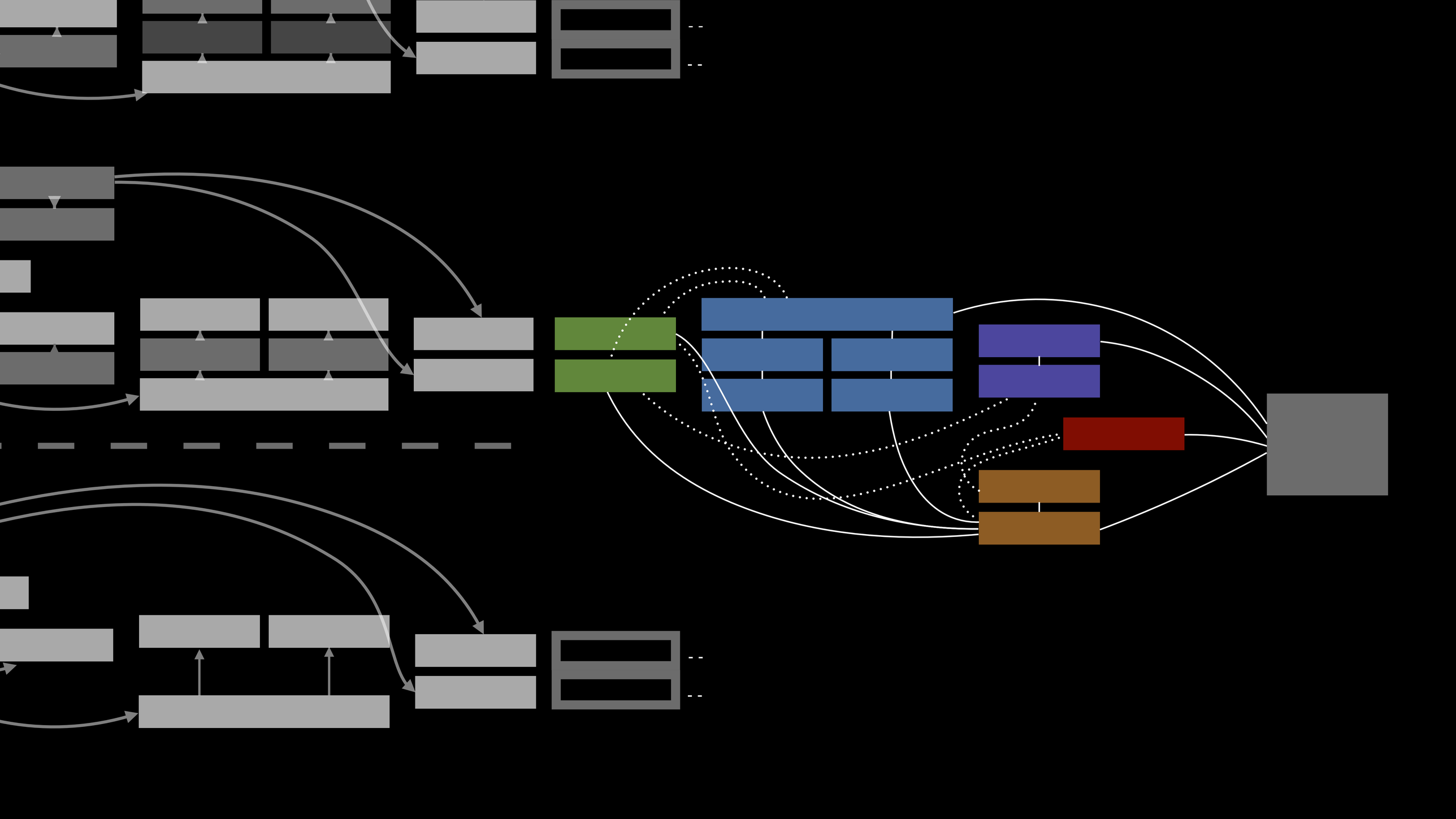

Reconciling &
Quoting
Reconciling &
Quoting

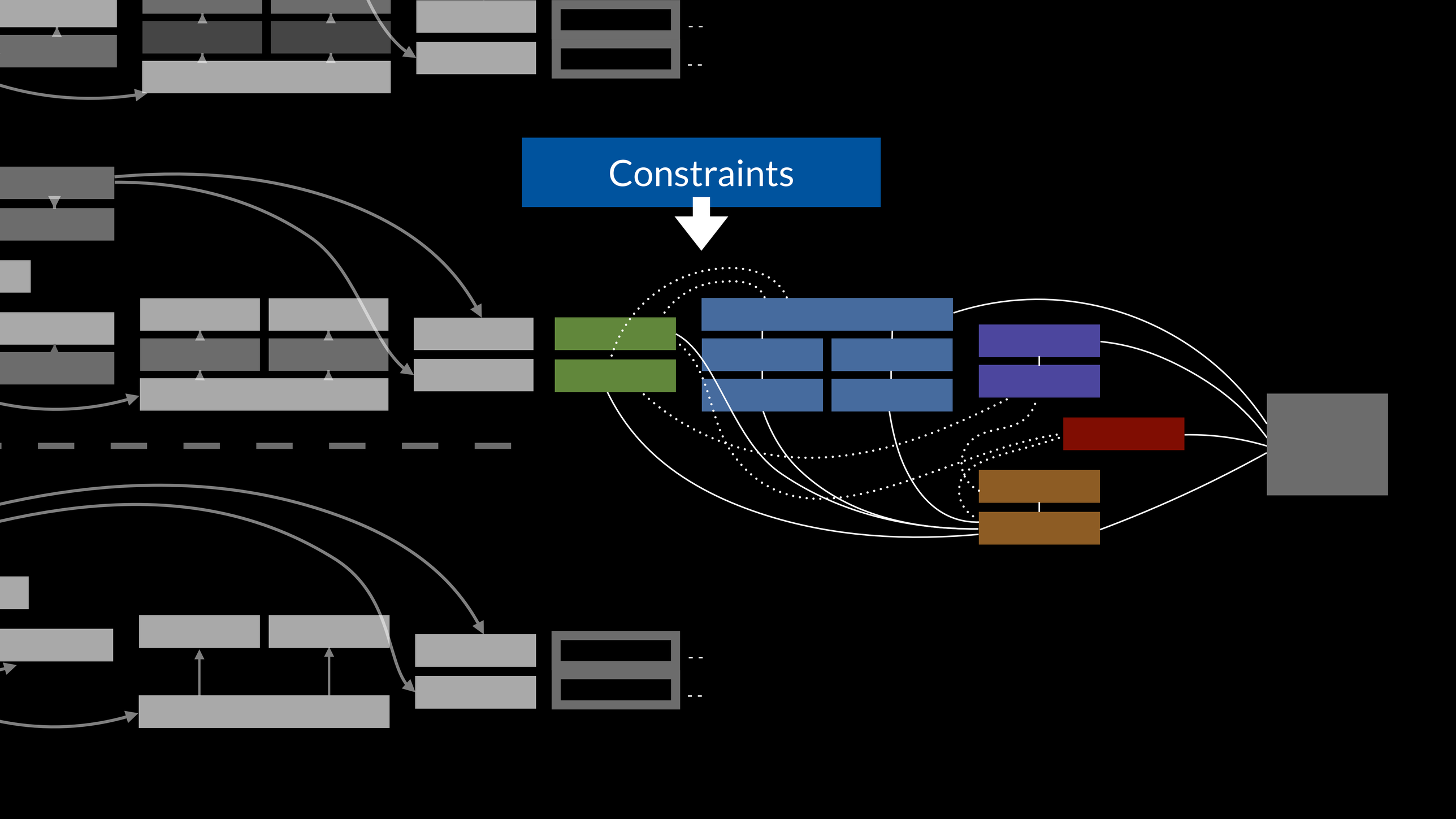

Reconciling &
Quoting
Incremental Lisp
macros
Reconciling &
Quoting

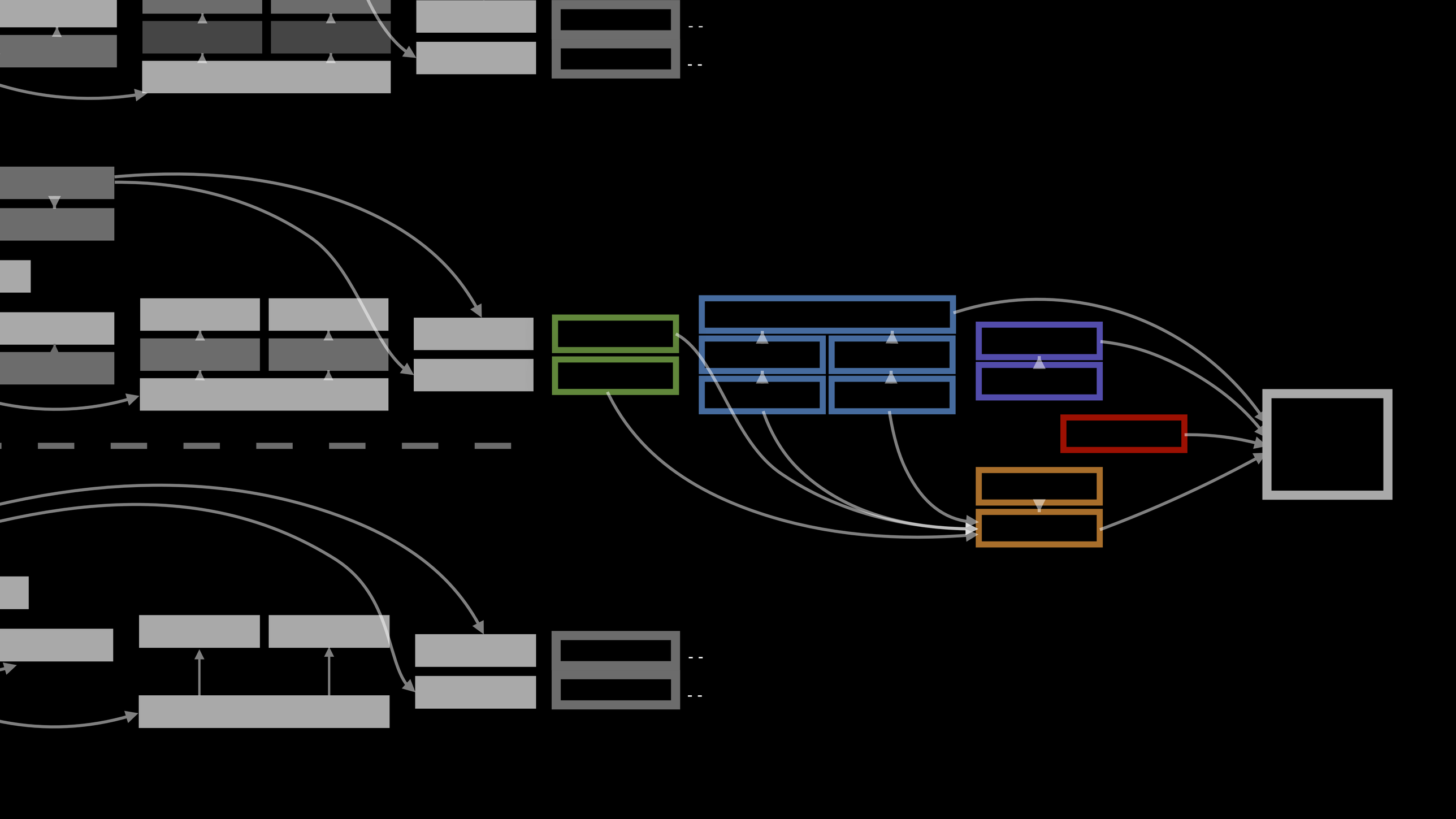


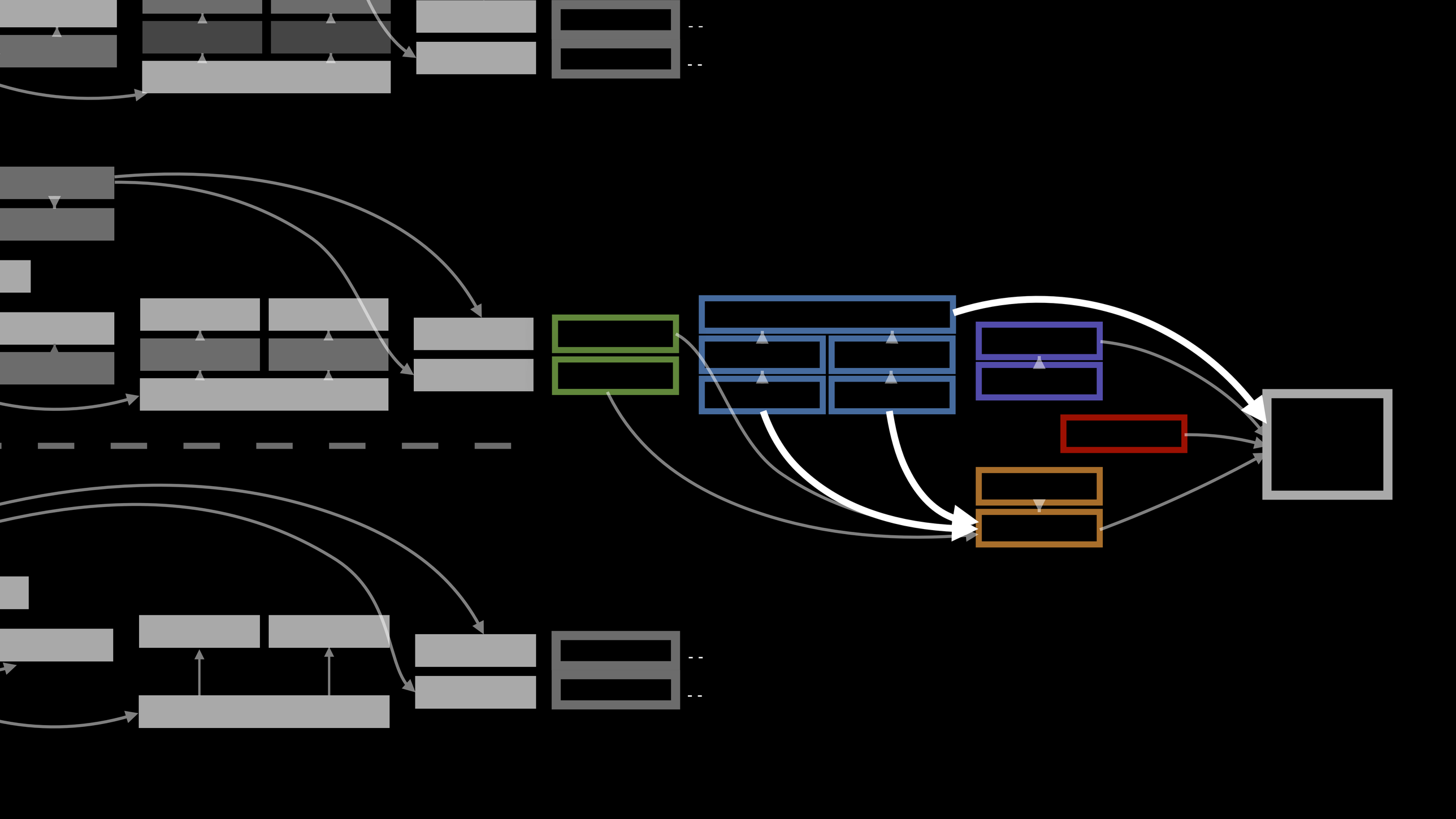


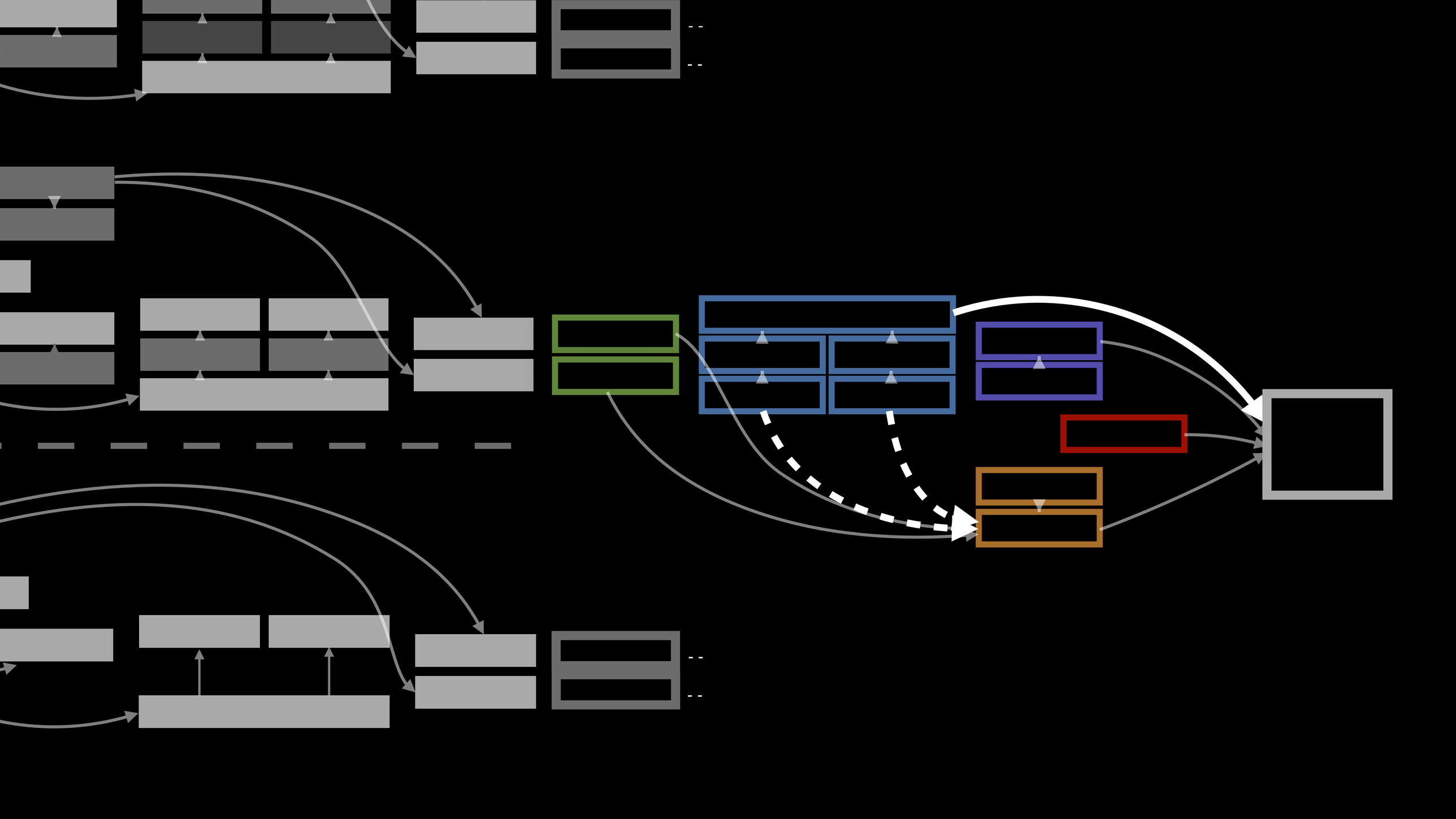


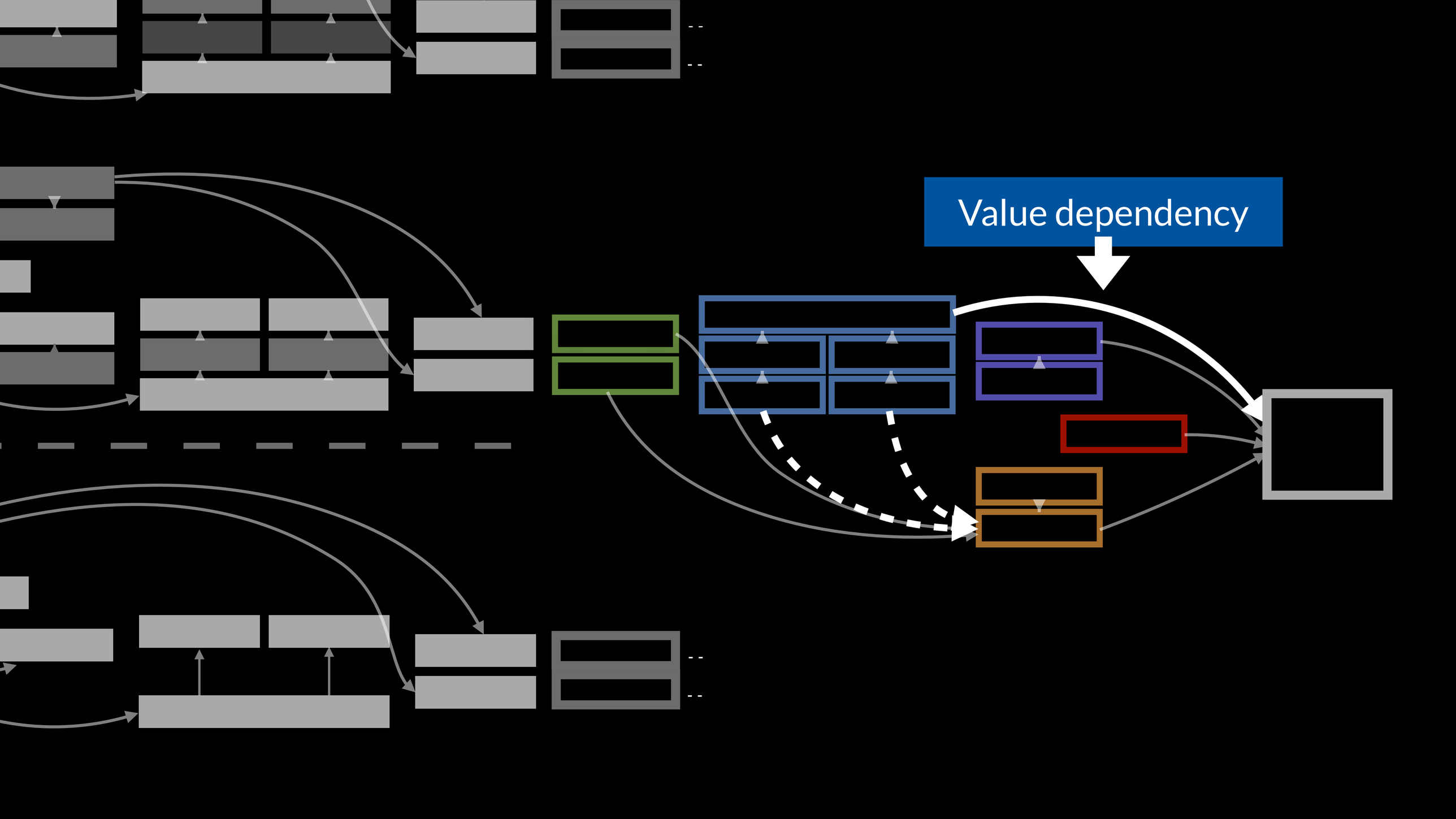


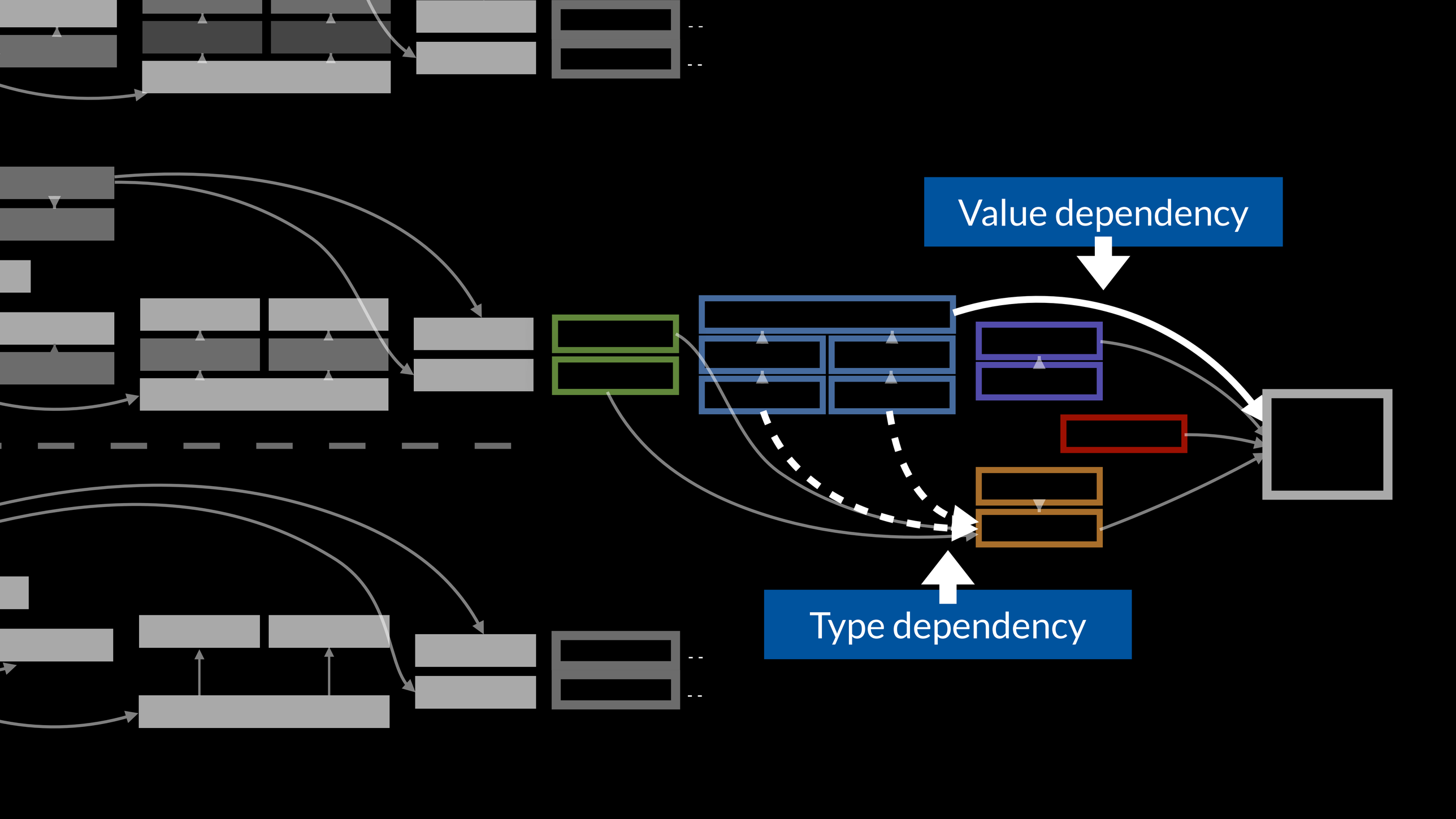

Constraints

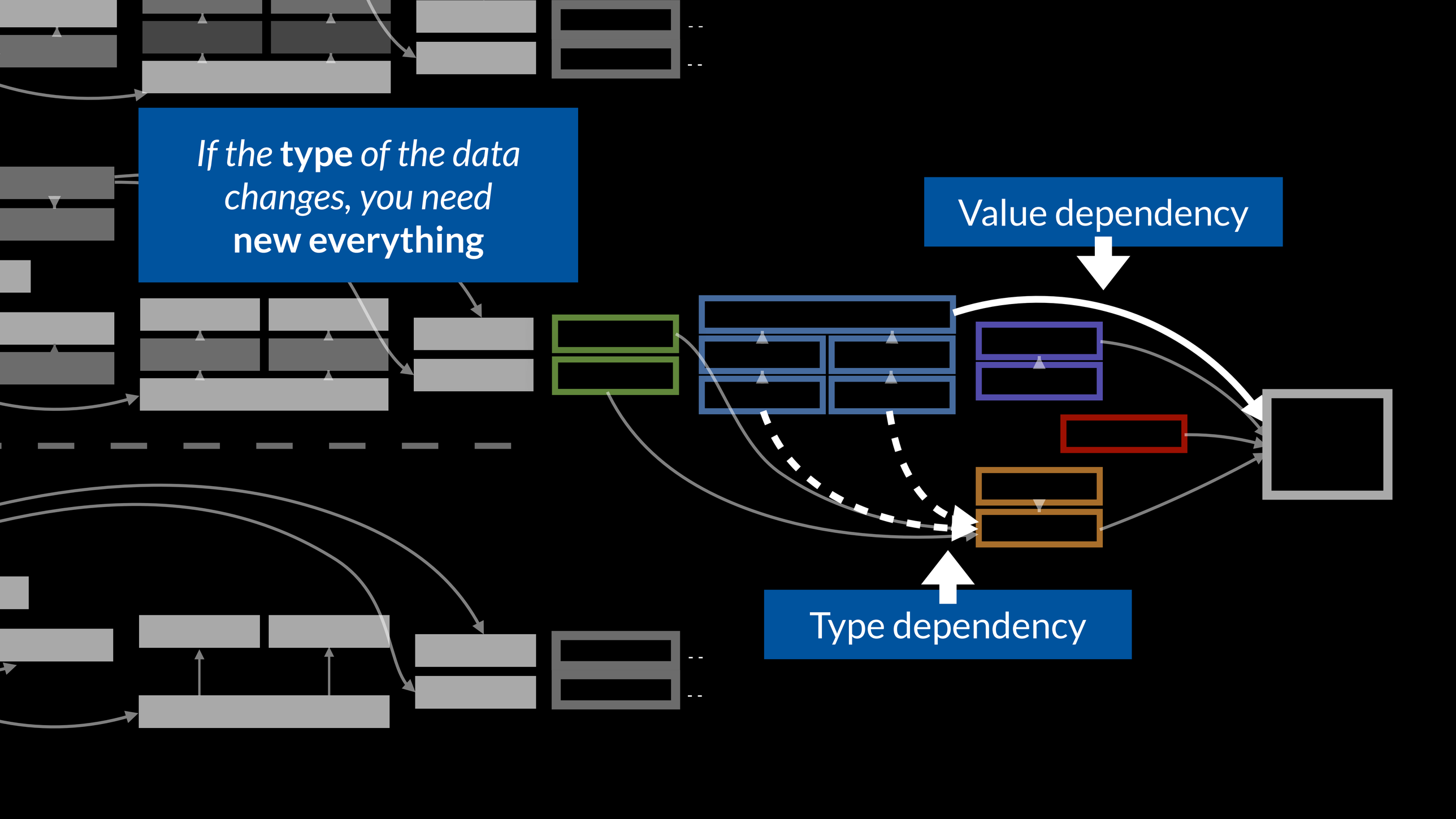


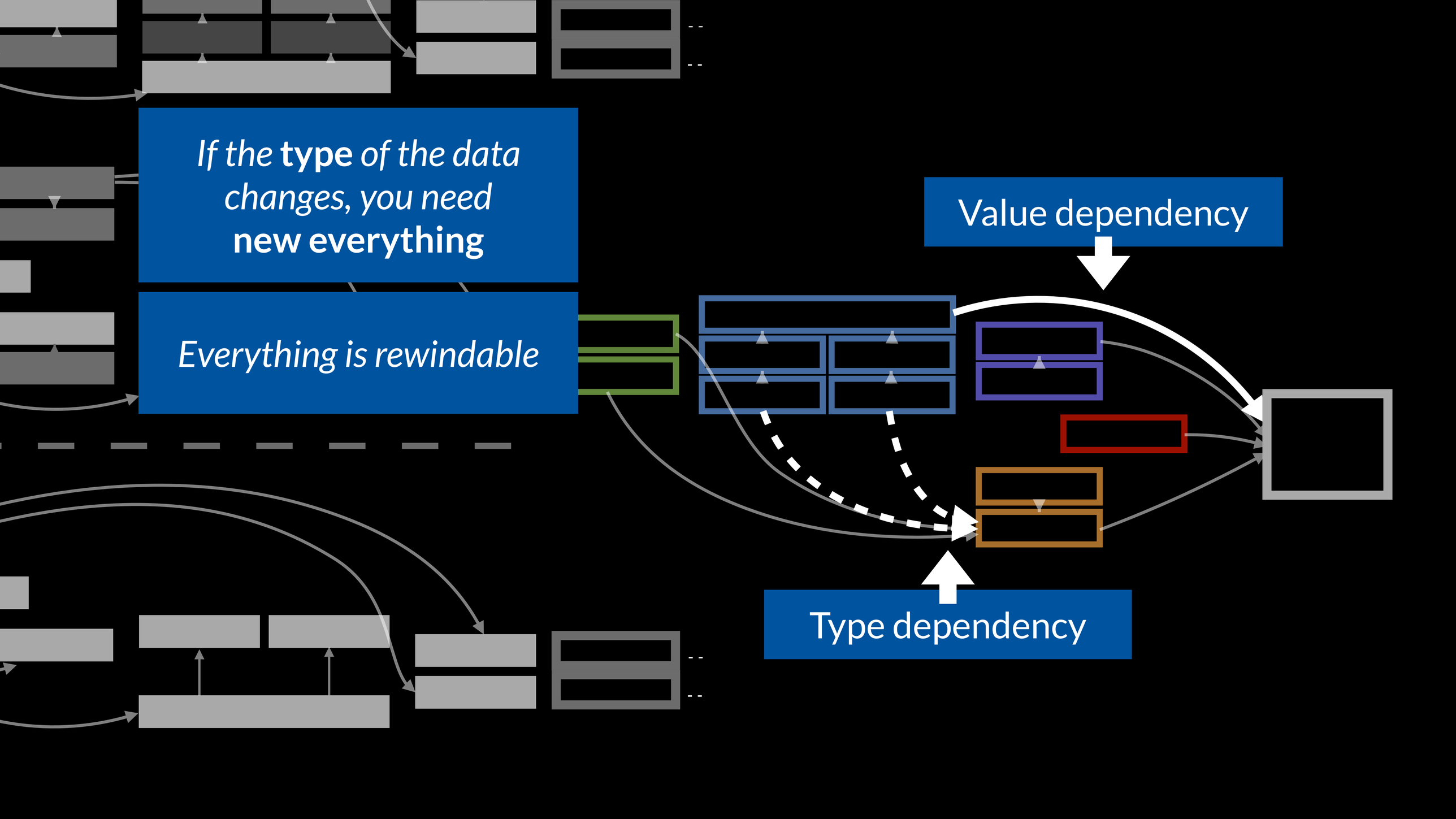


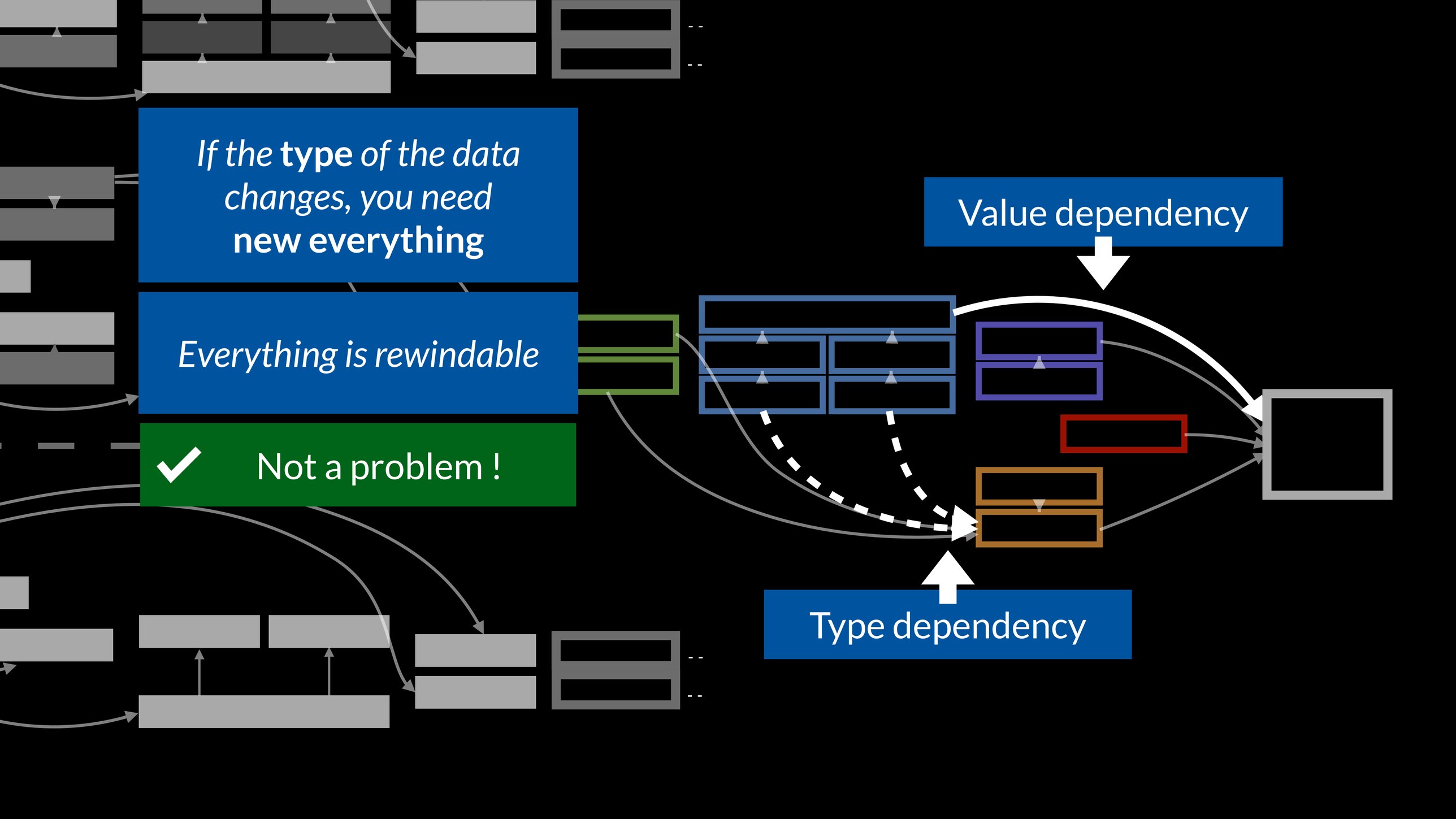


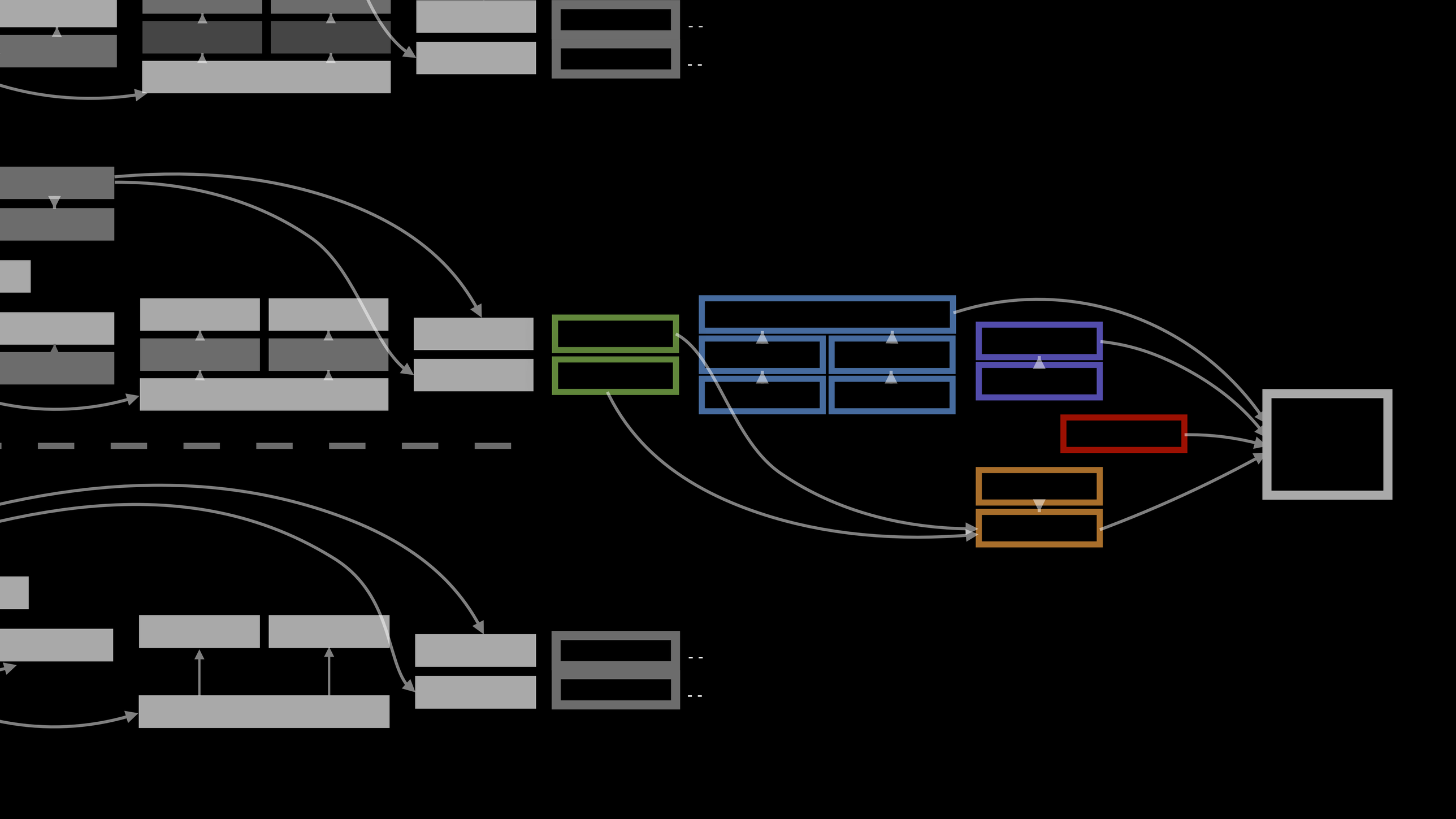

Value dependency

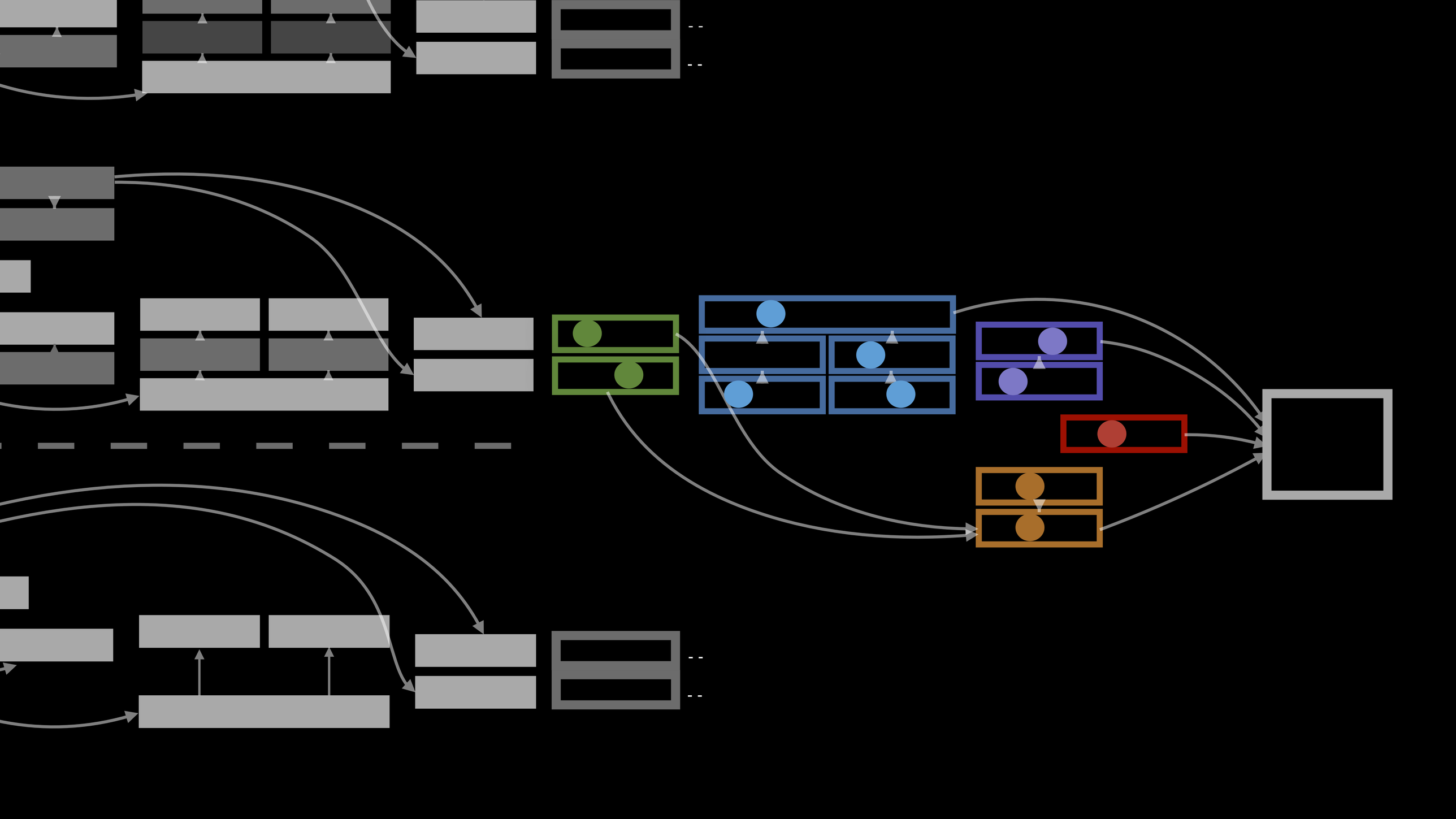

Value dependency
Type dependency

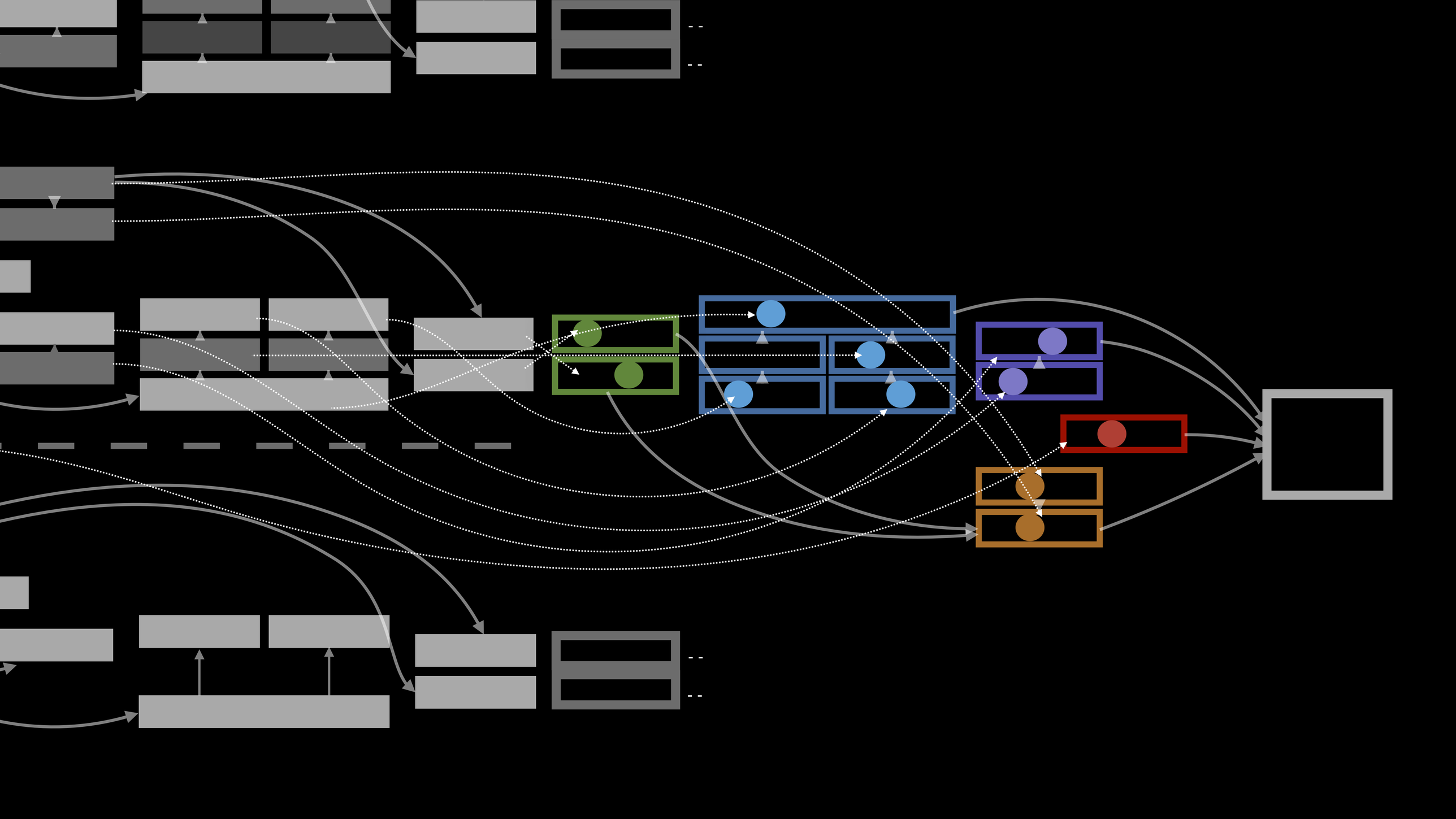

If the **type** *of the data changes, you need* **new everything**
Value dependency
Type dependency

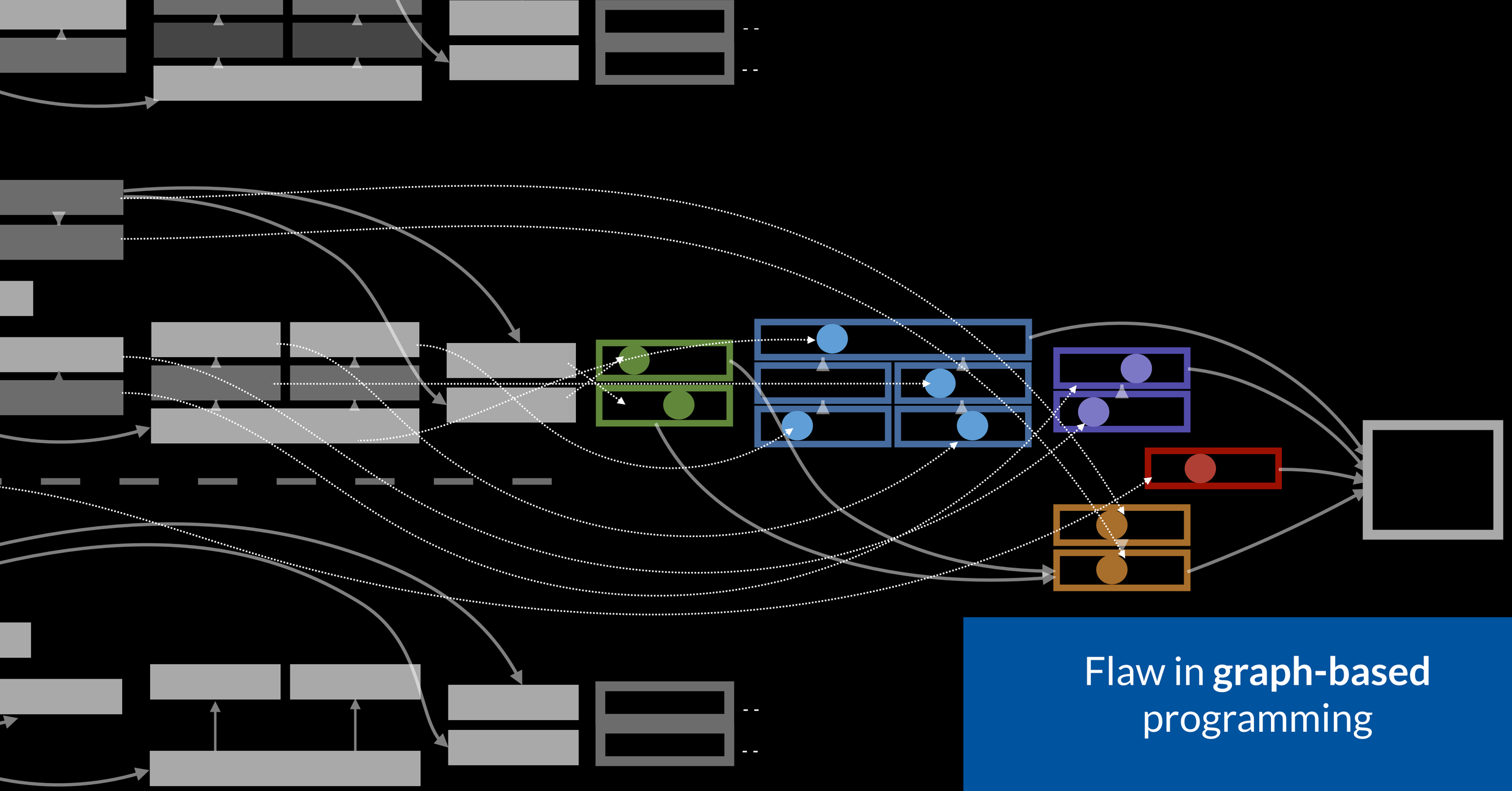

If the **type** *of the data changes, you need* **new everything**
Everything is rewindable
Value dependency
Type dependency


If the **type** *of the data changes, you need* **new everything**
Everything is rewindable
✔ Not a problem !
Value dependency
Type dependency



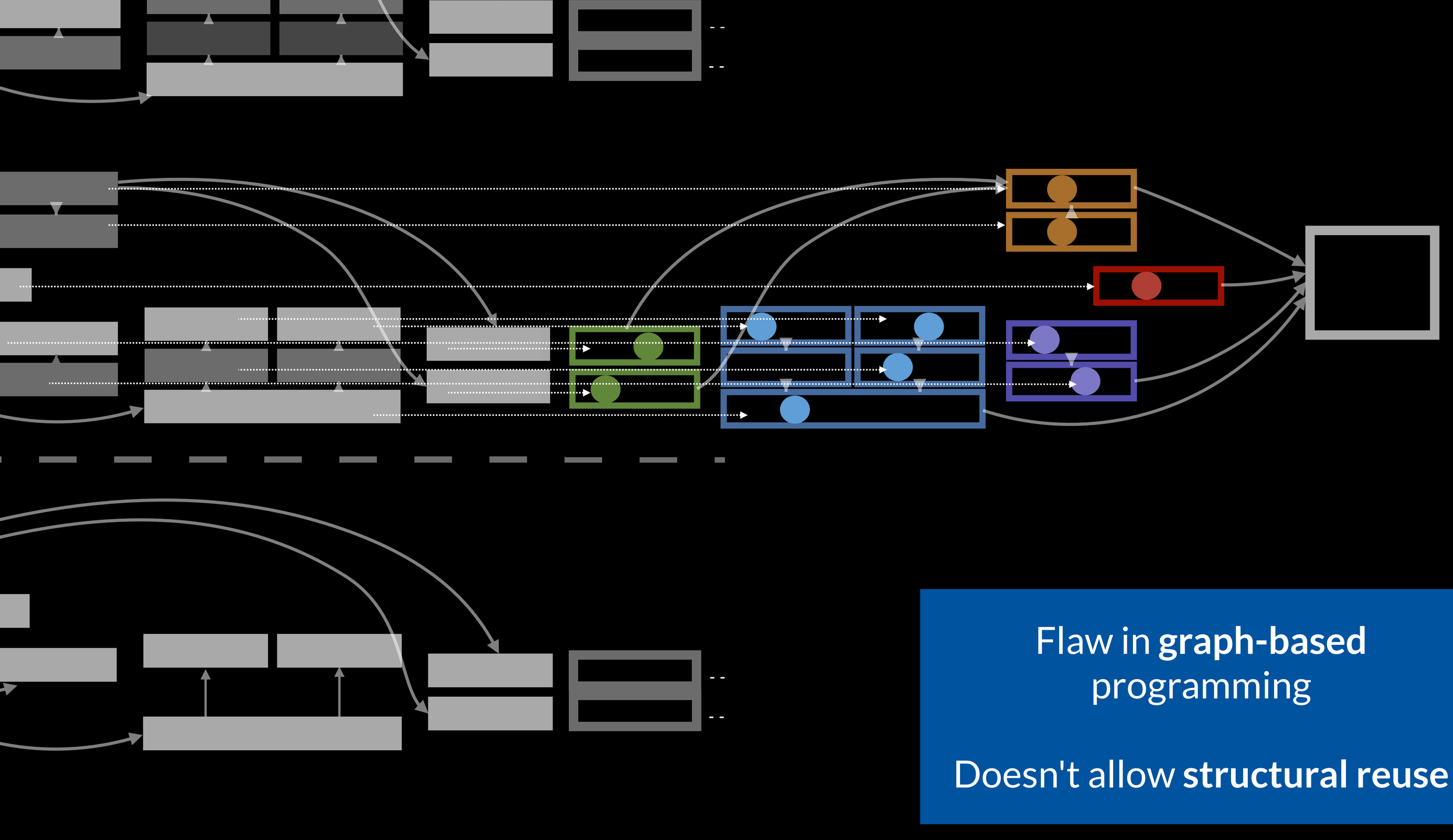






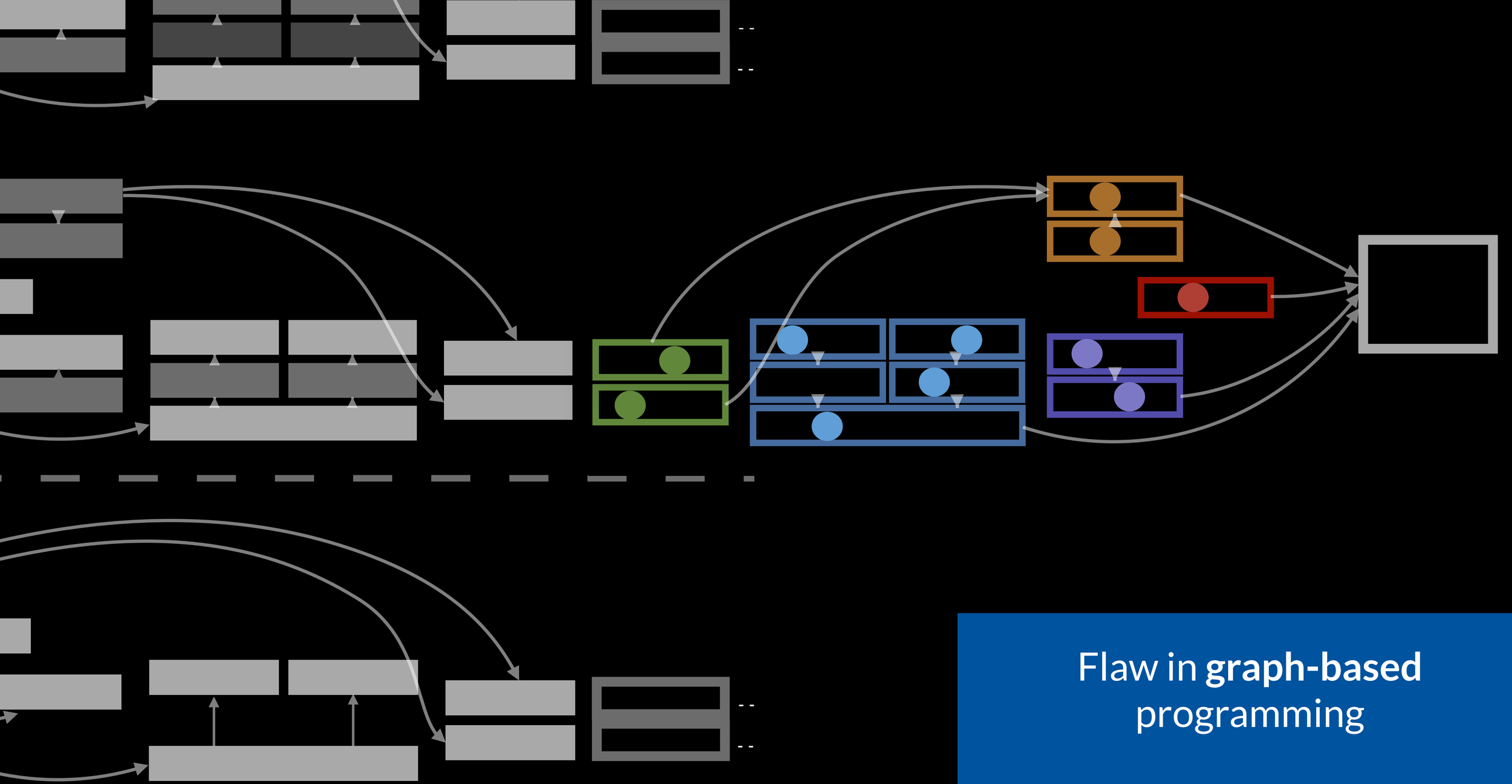
Flaw in **graph-based** programming

Doesn't allow **structural reuse**



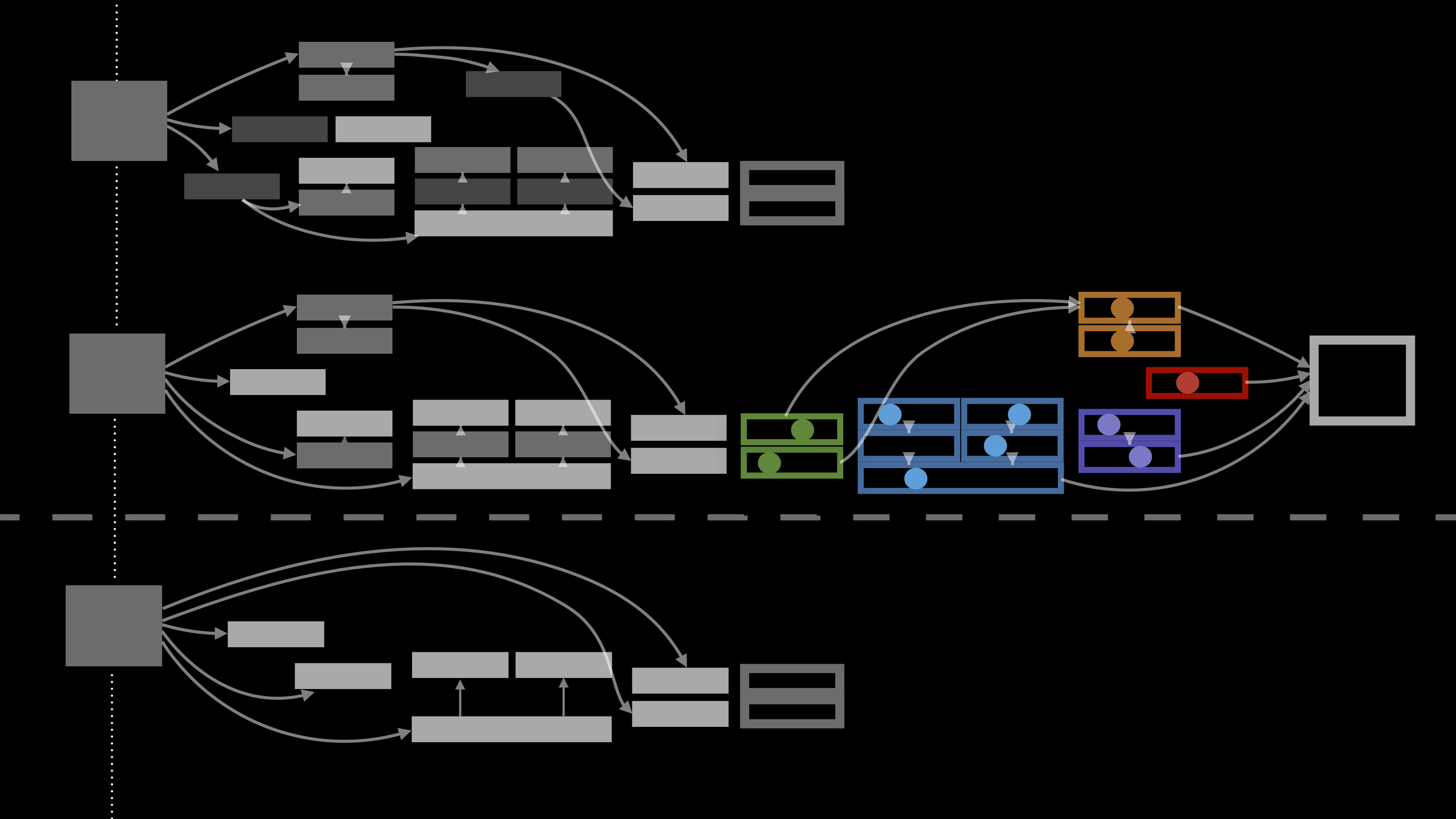
Flaw in **graph-based** programming

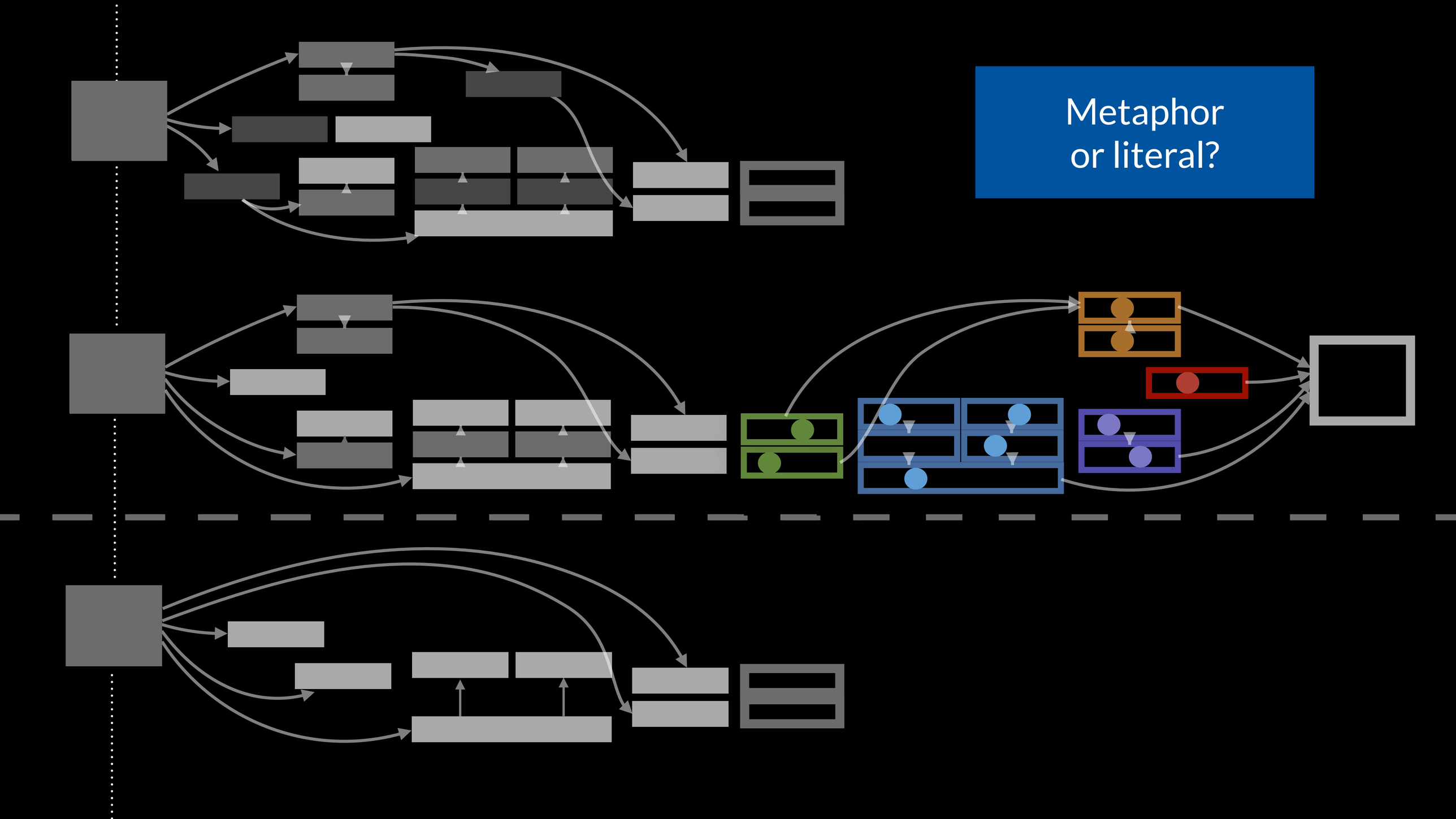
Doesn't allow **structural reuse**

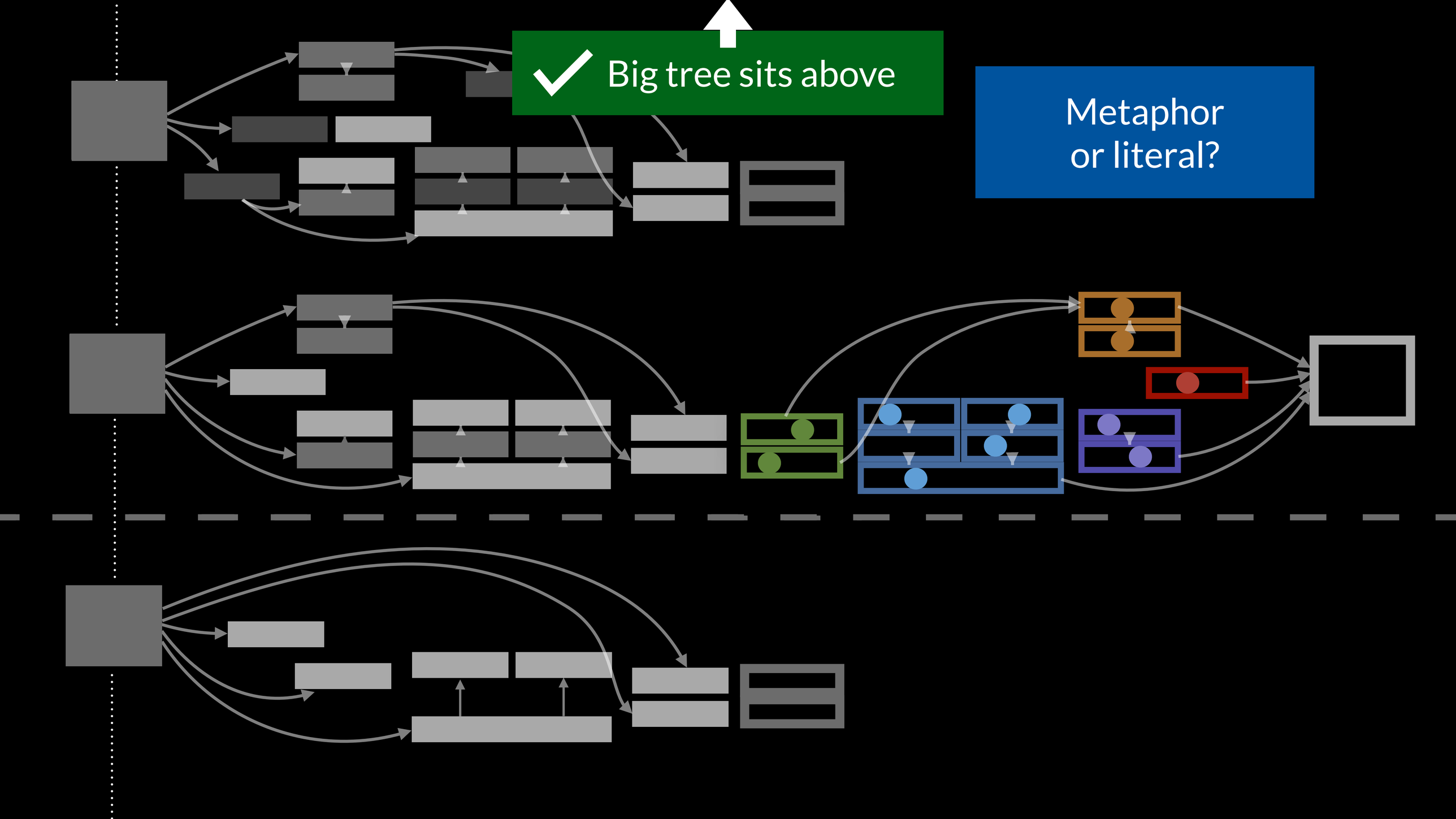


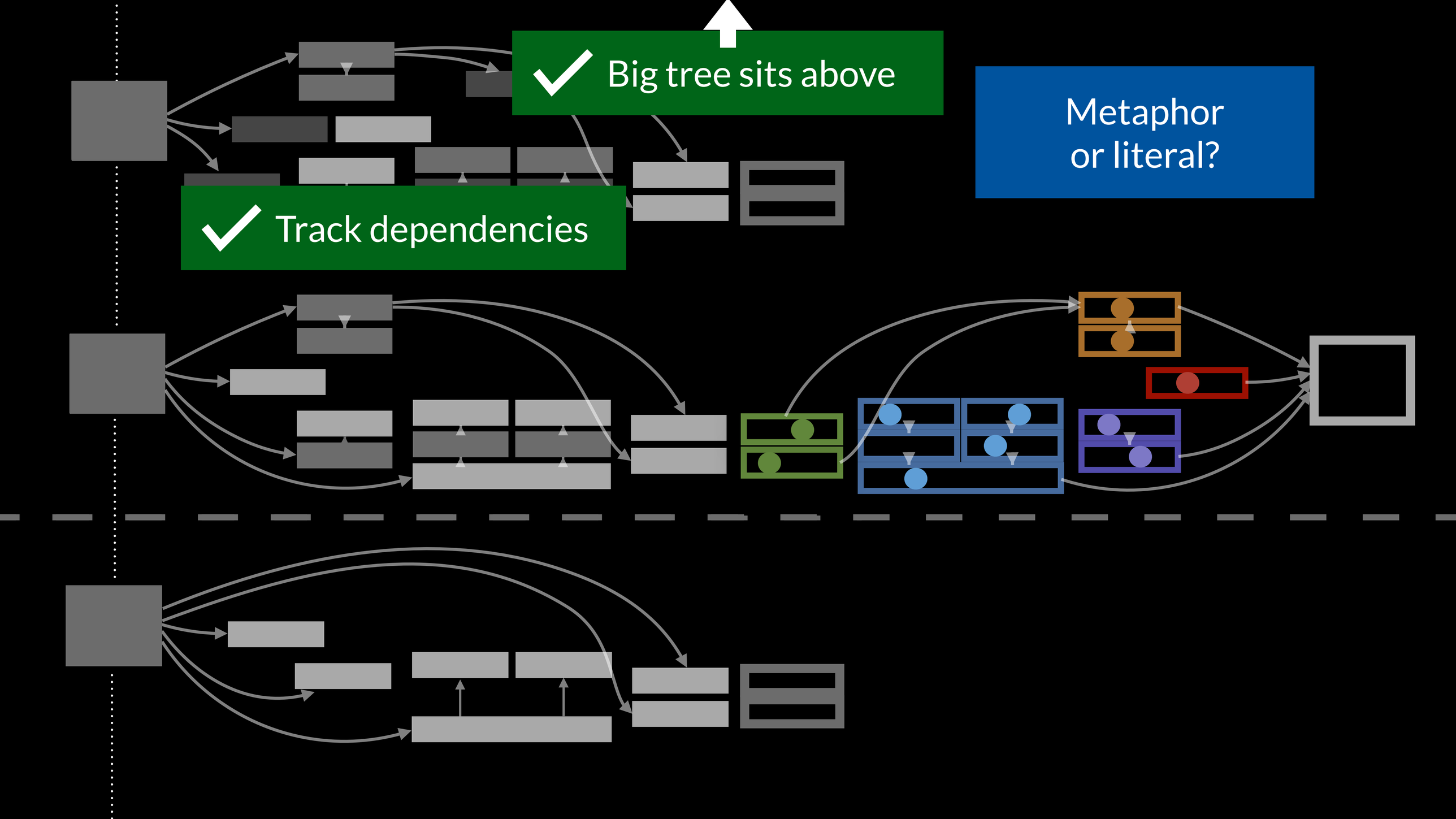
Flaw in **graph-based** programming

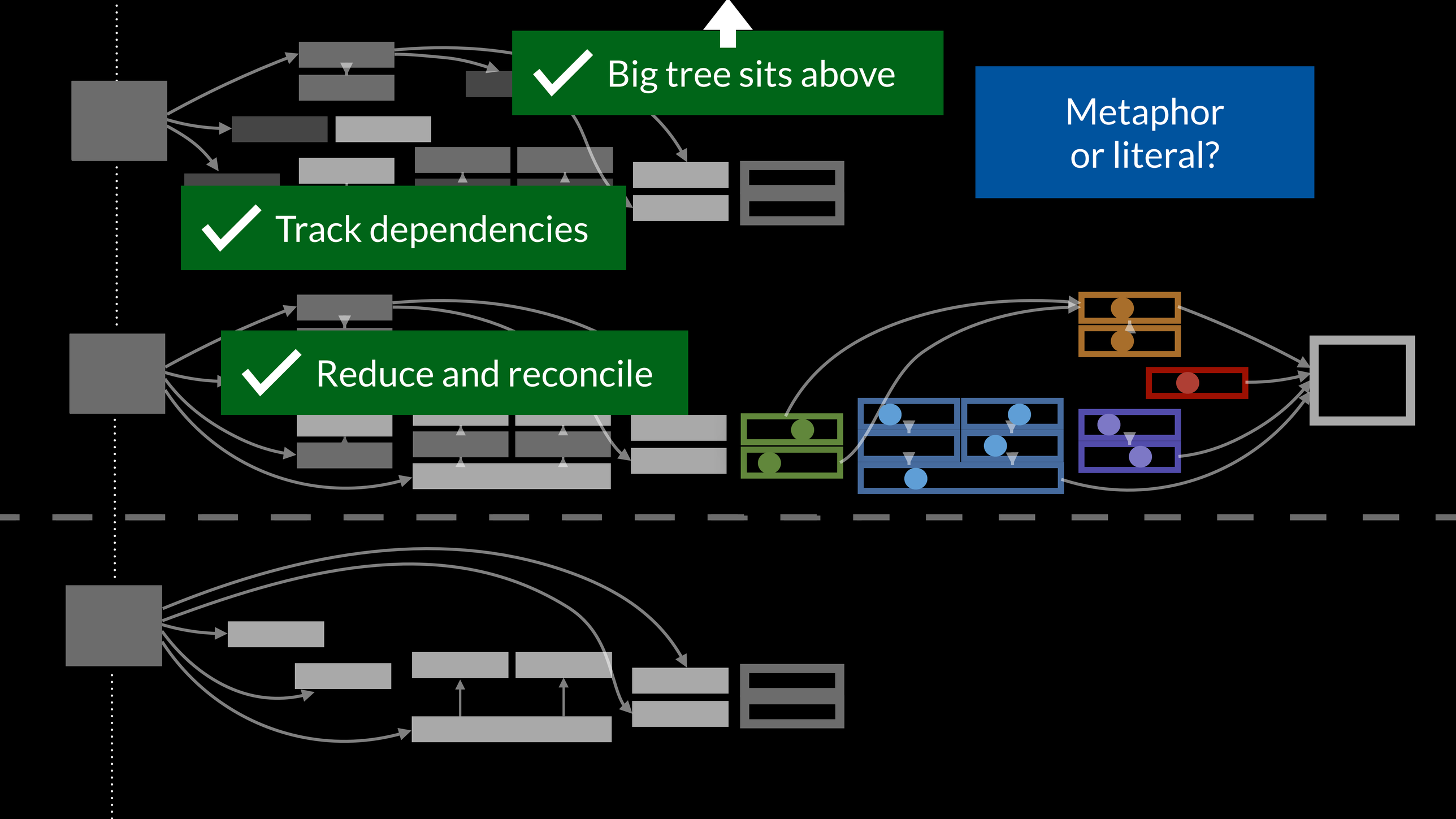
Doesn't allow **structural reuse**

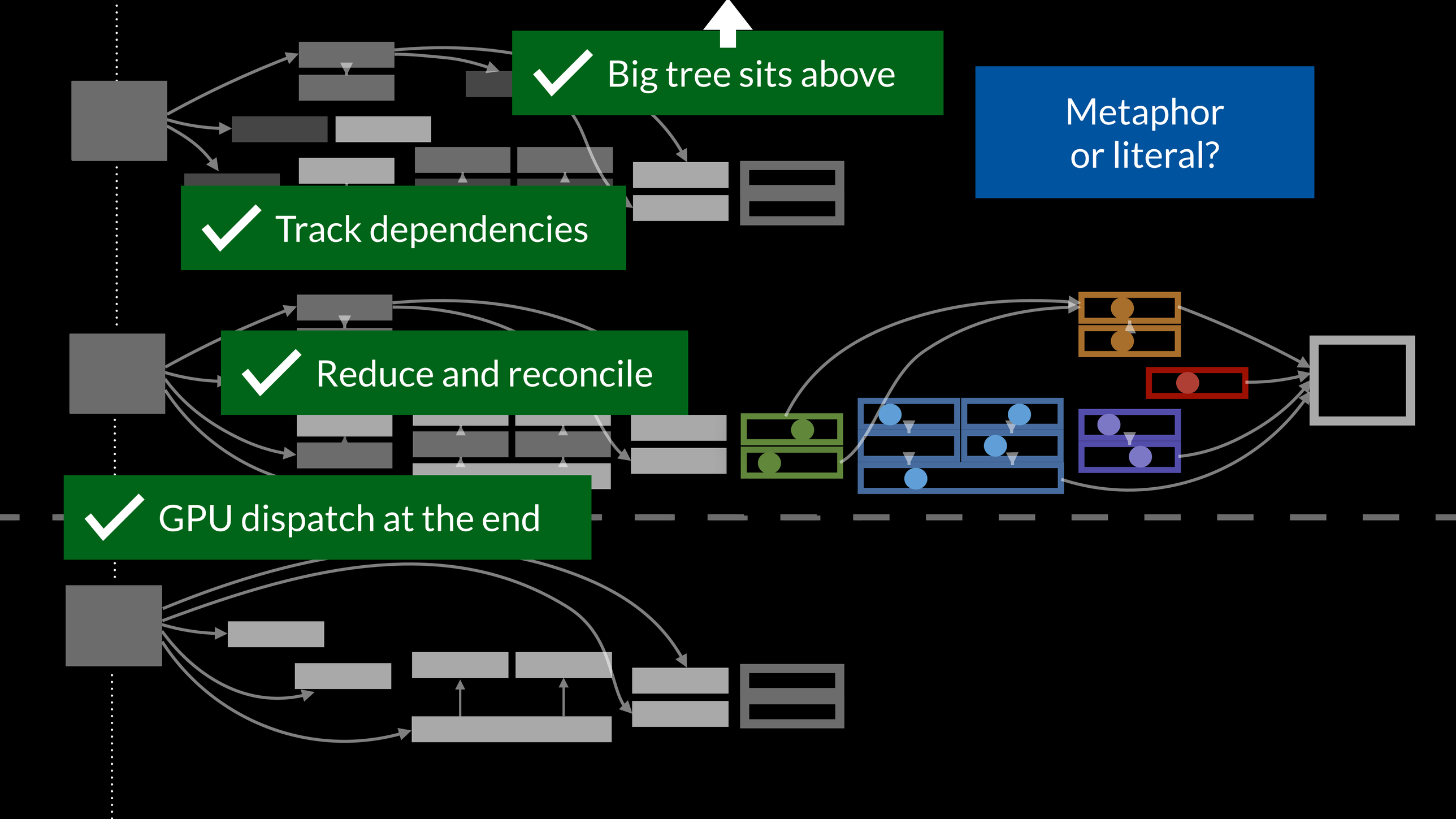


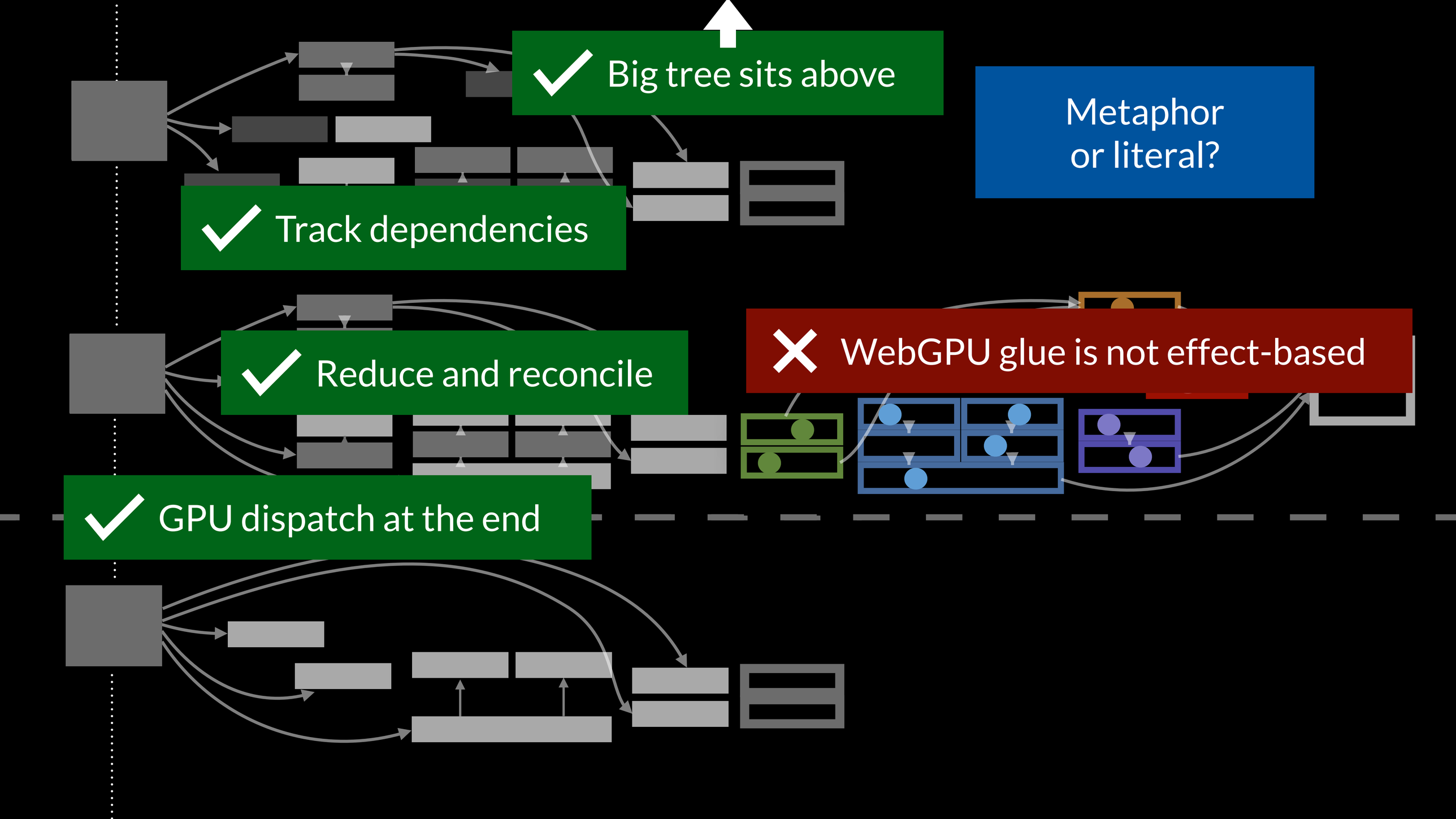

Metaphor
or literal?

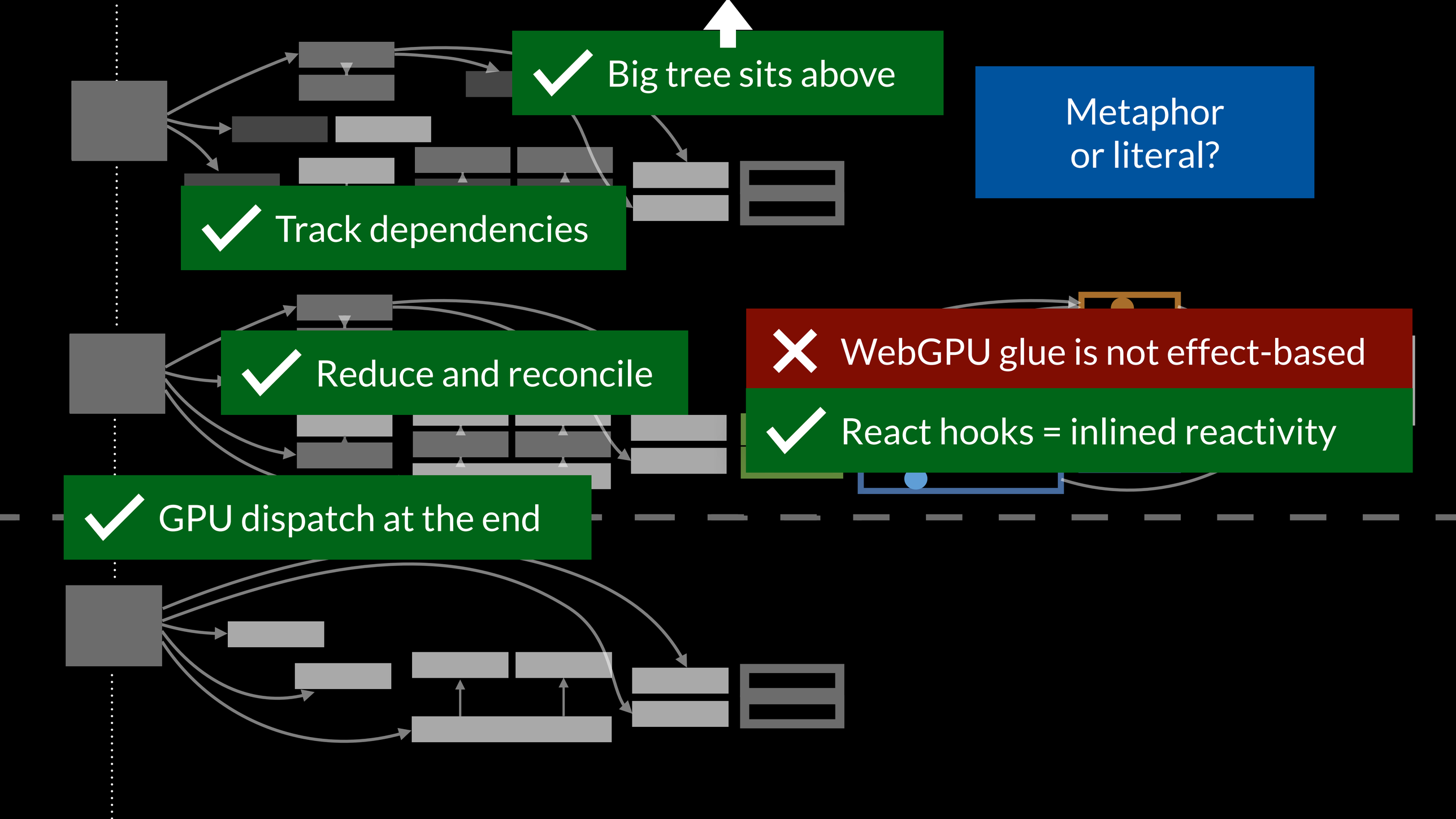

Big tree sits above
Metaphor
or literal?

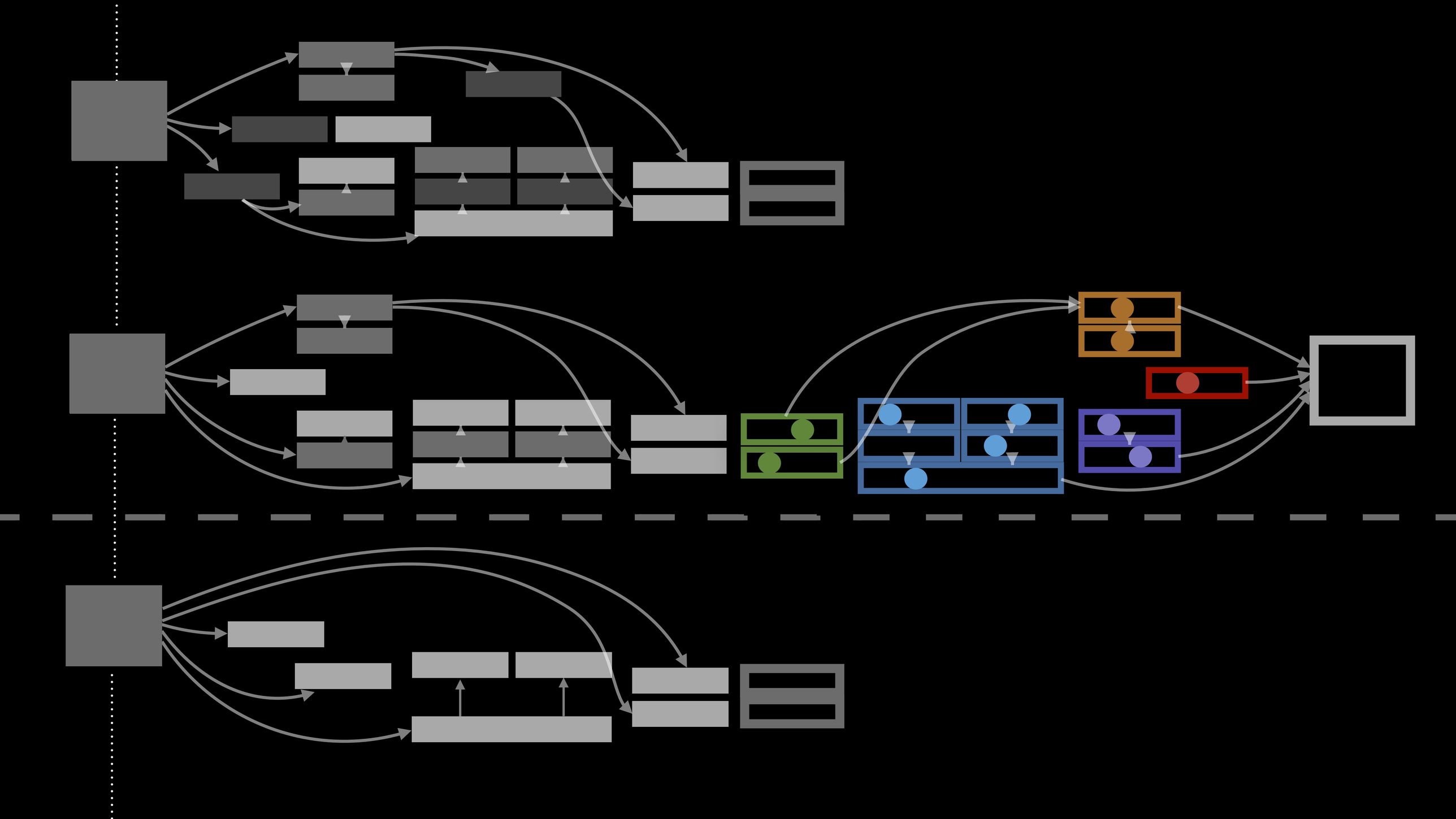

Big tree sits above
Metaphor
or literal?
Track dependencies


Big tree sits above
Metaphor
or literal?
Track dependencies
Reduce and reconcile

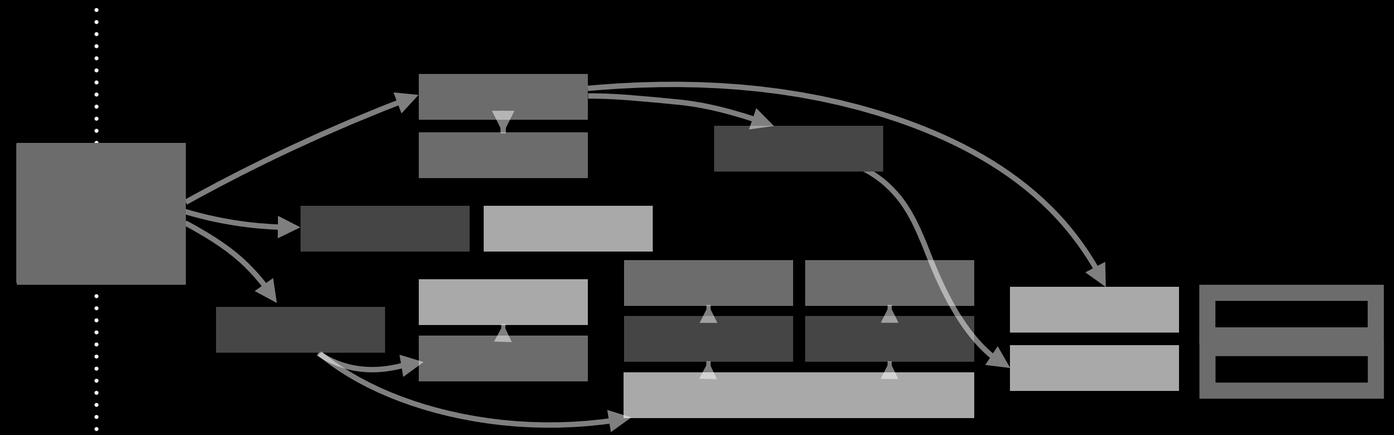

Big tree sits above
Metaphor
or literal?
Track dependencies
Reduce and reconcile
GPU dispatch at the end


Big tree sits above
Metaphor
or literal?
Track dependencies
Reduce and reconcile
WebGPU glue is not effect-based
GPU dispatch at the end

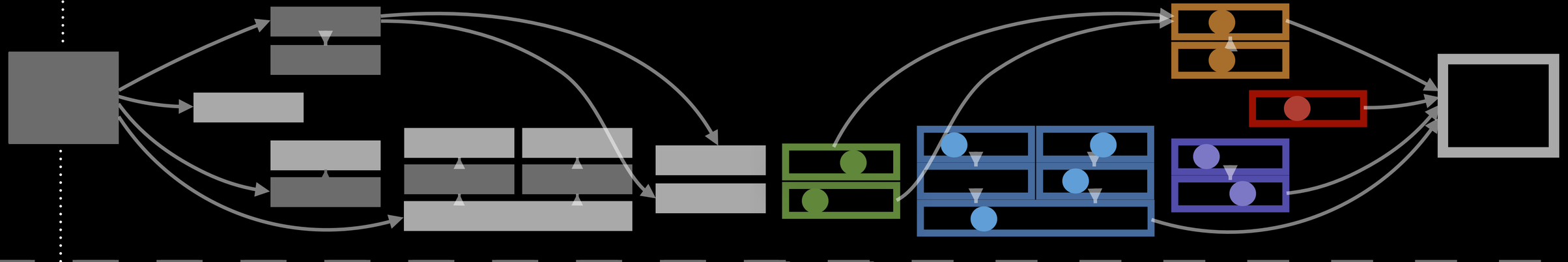

Big tree sits above
Metaphor
or literal?
Track dependencies
Reduce and reconcile
WebGPU glue is not effect-based
React hooks = inlined reactivity
GPU dispatch at the end

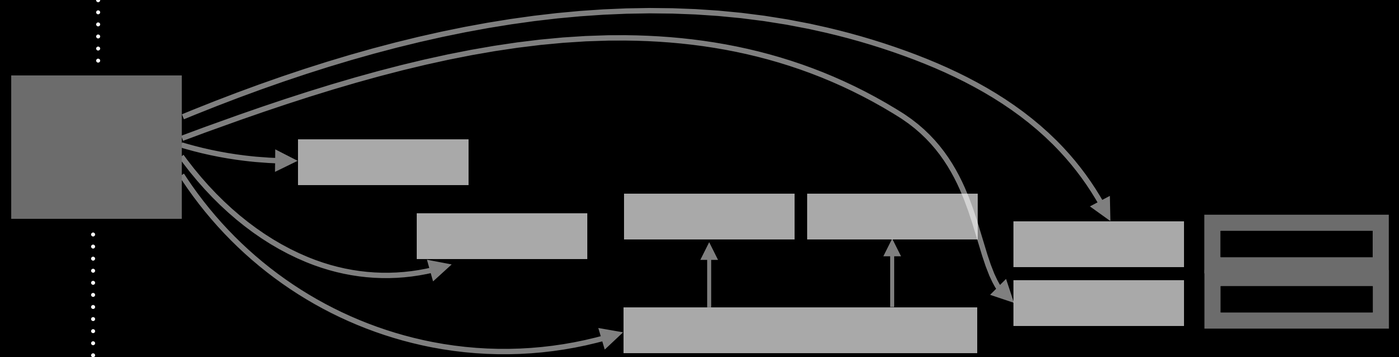



Data shapes the **code**
that **generates** it

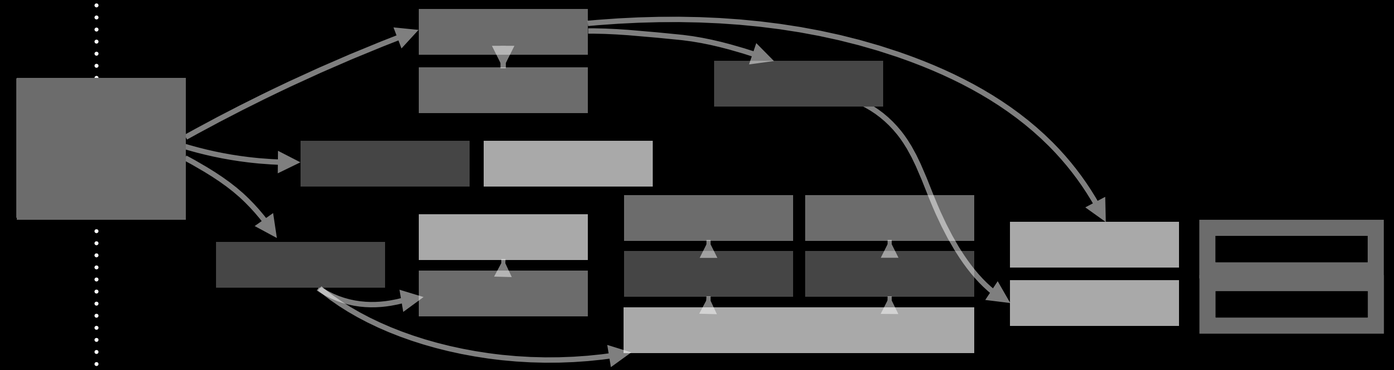


Data shapes the **code** that **generates** it

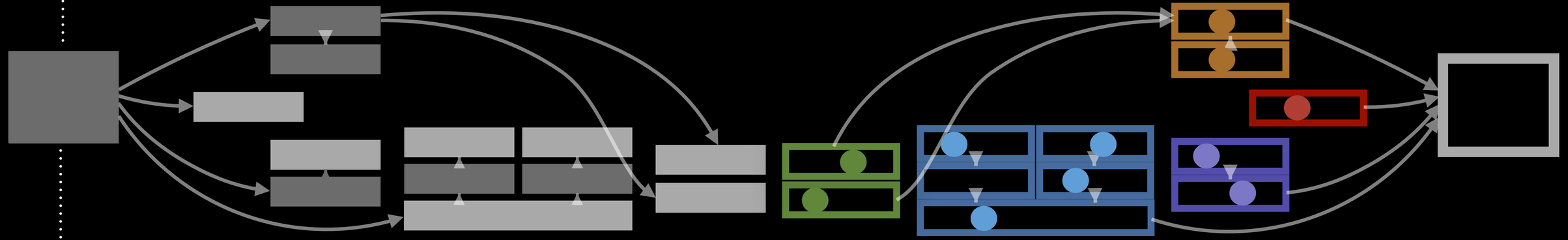




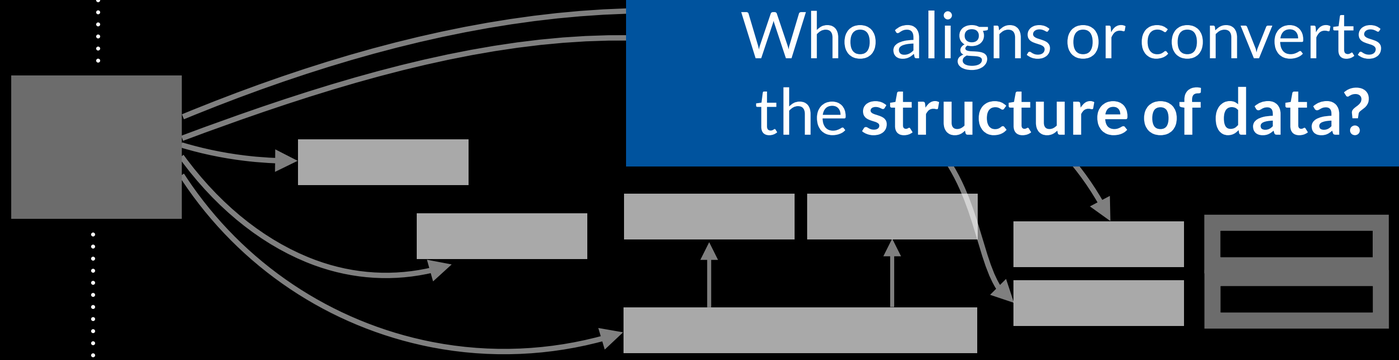
Data shapes the **code** that **consumes** it



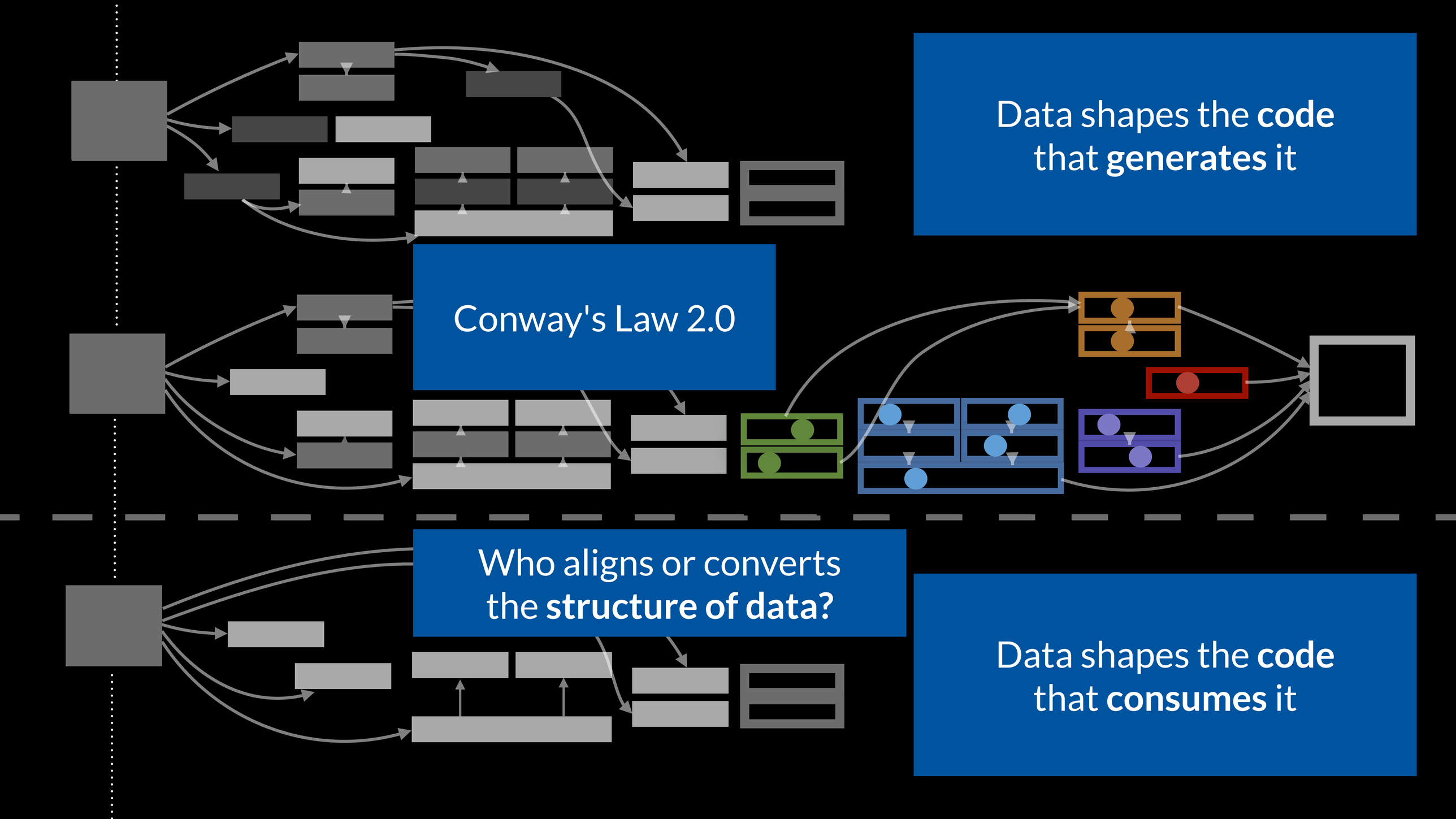
Data shapes the **code** that **generates** it

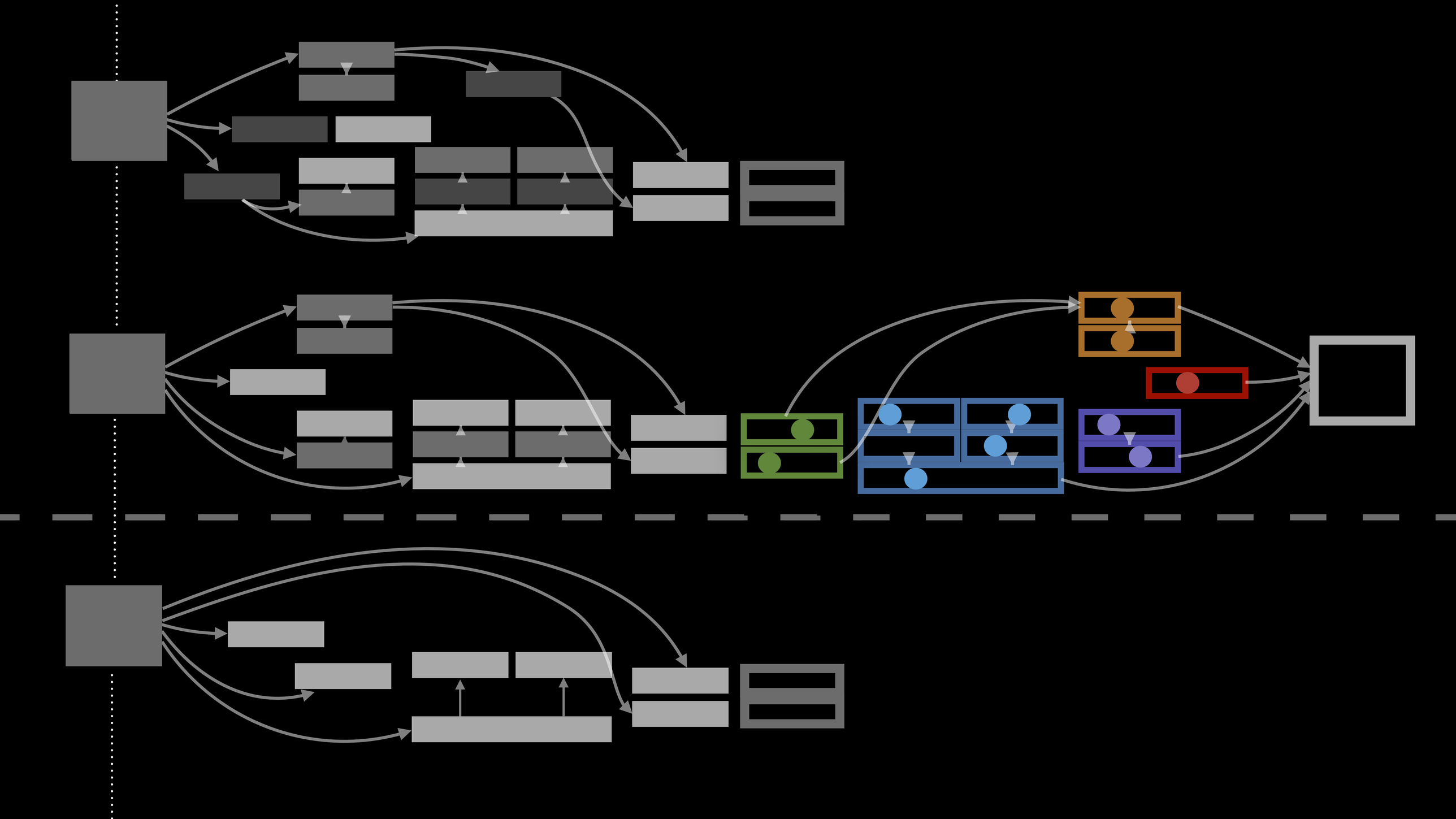


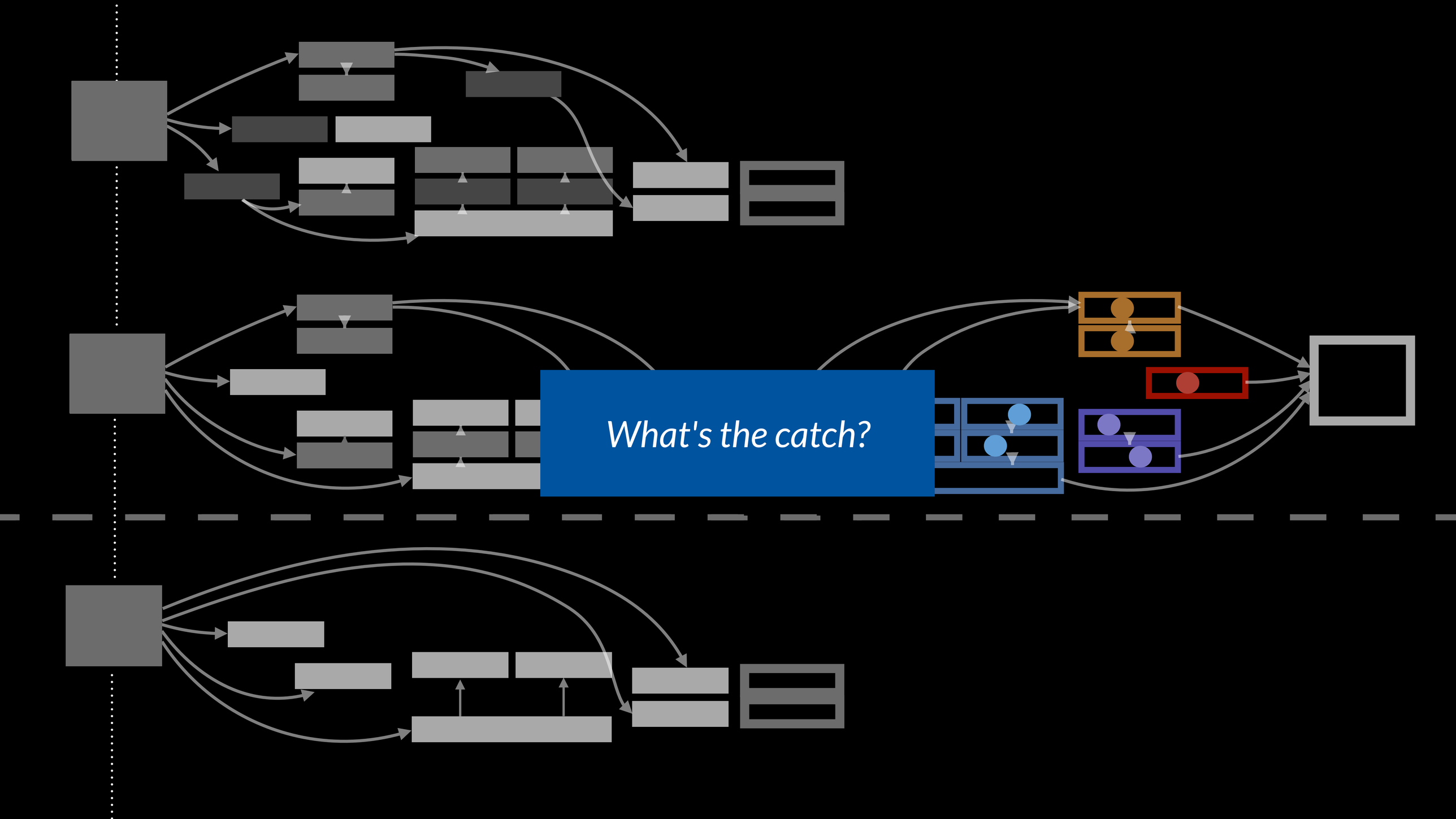
Who aligns or converts the **structure of data?**

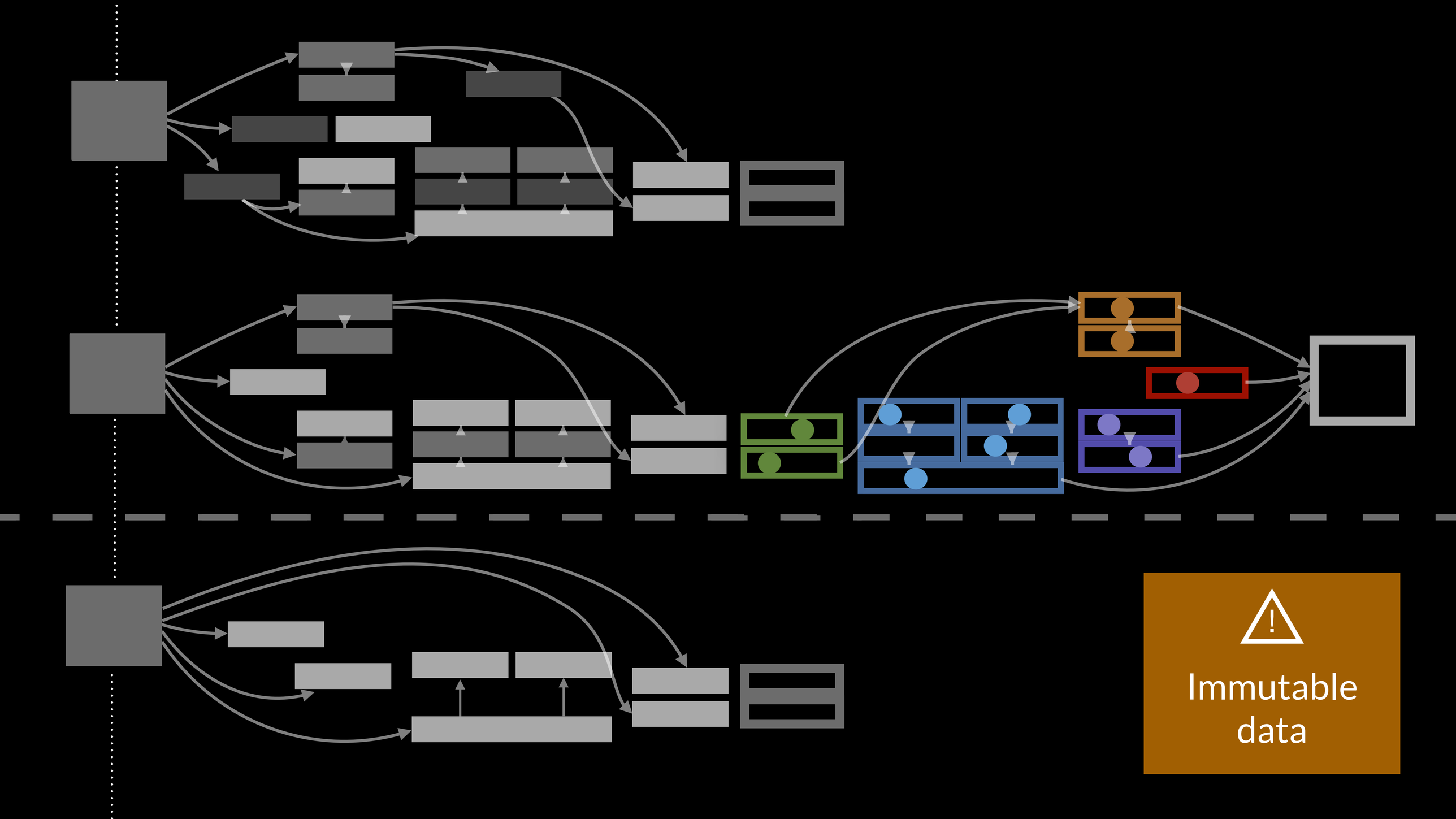


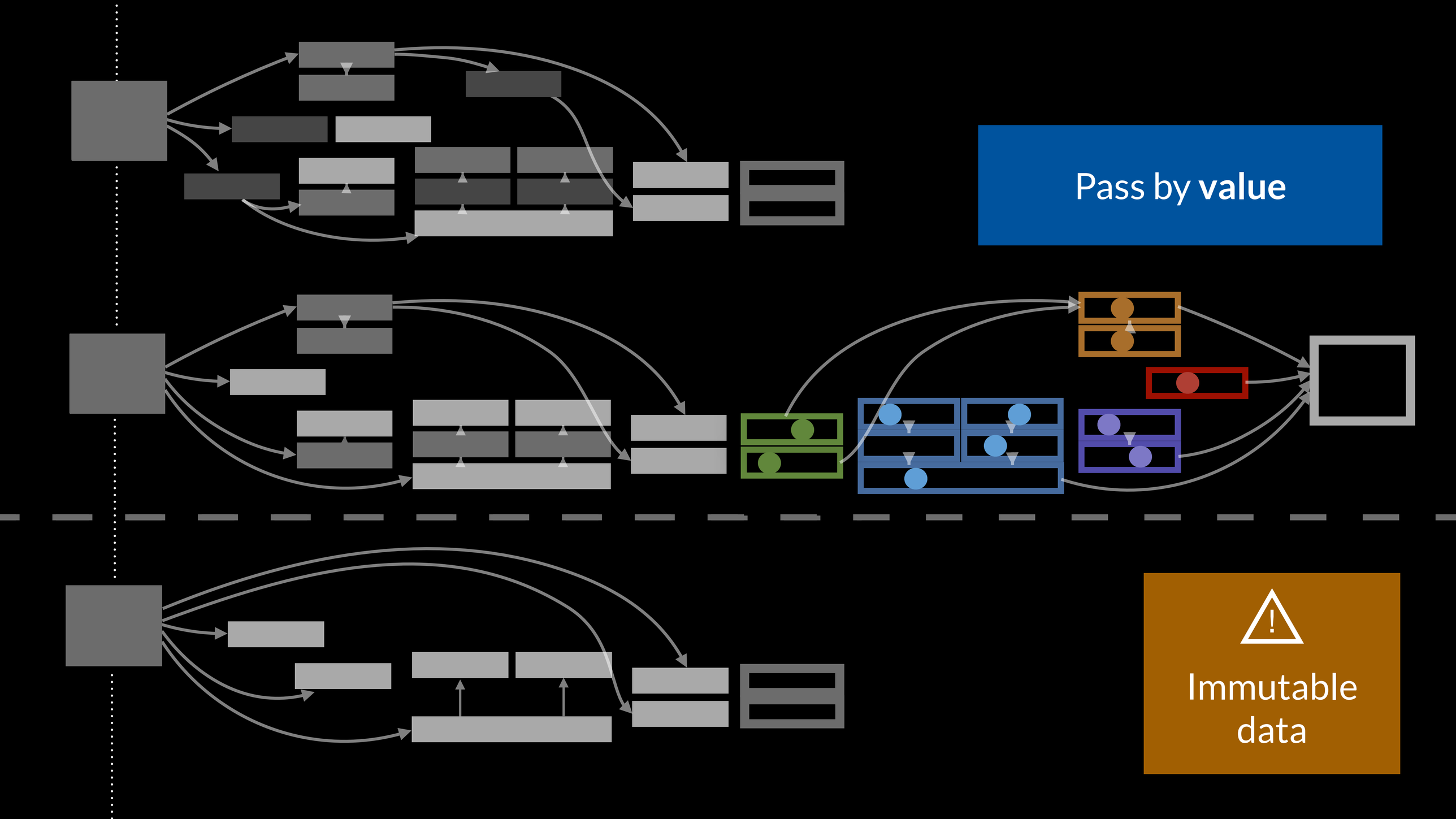
Data shapes the **code** that **consumes** it

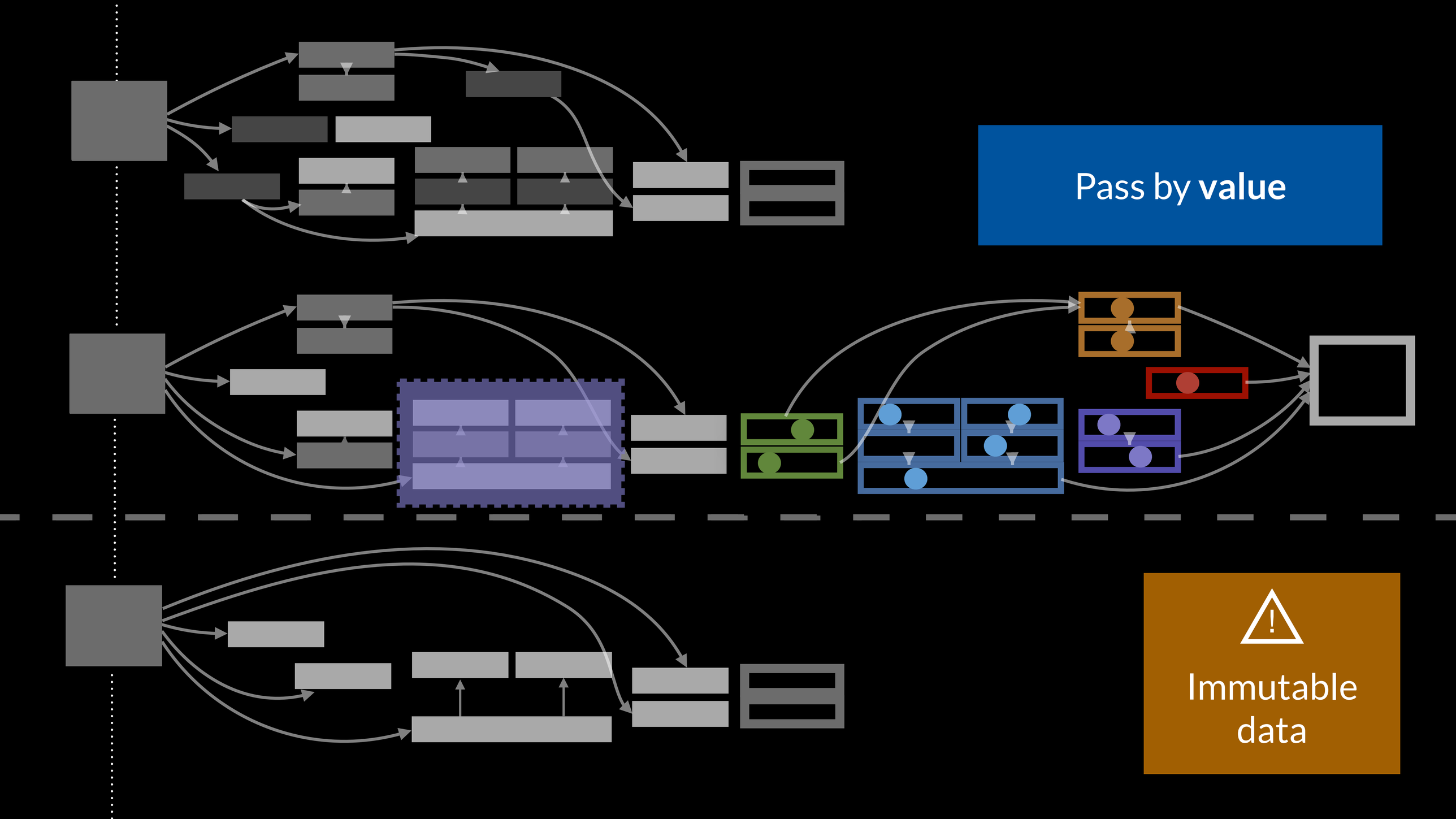

Data shapes the **code** that **generates** it
Conway's Law 2.0
Who aligns or converts the **structure of data?**
Data shapes the **code** that **consumes** it

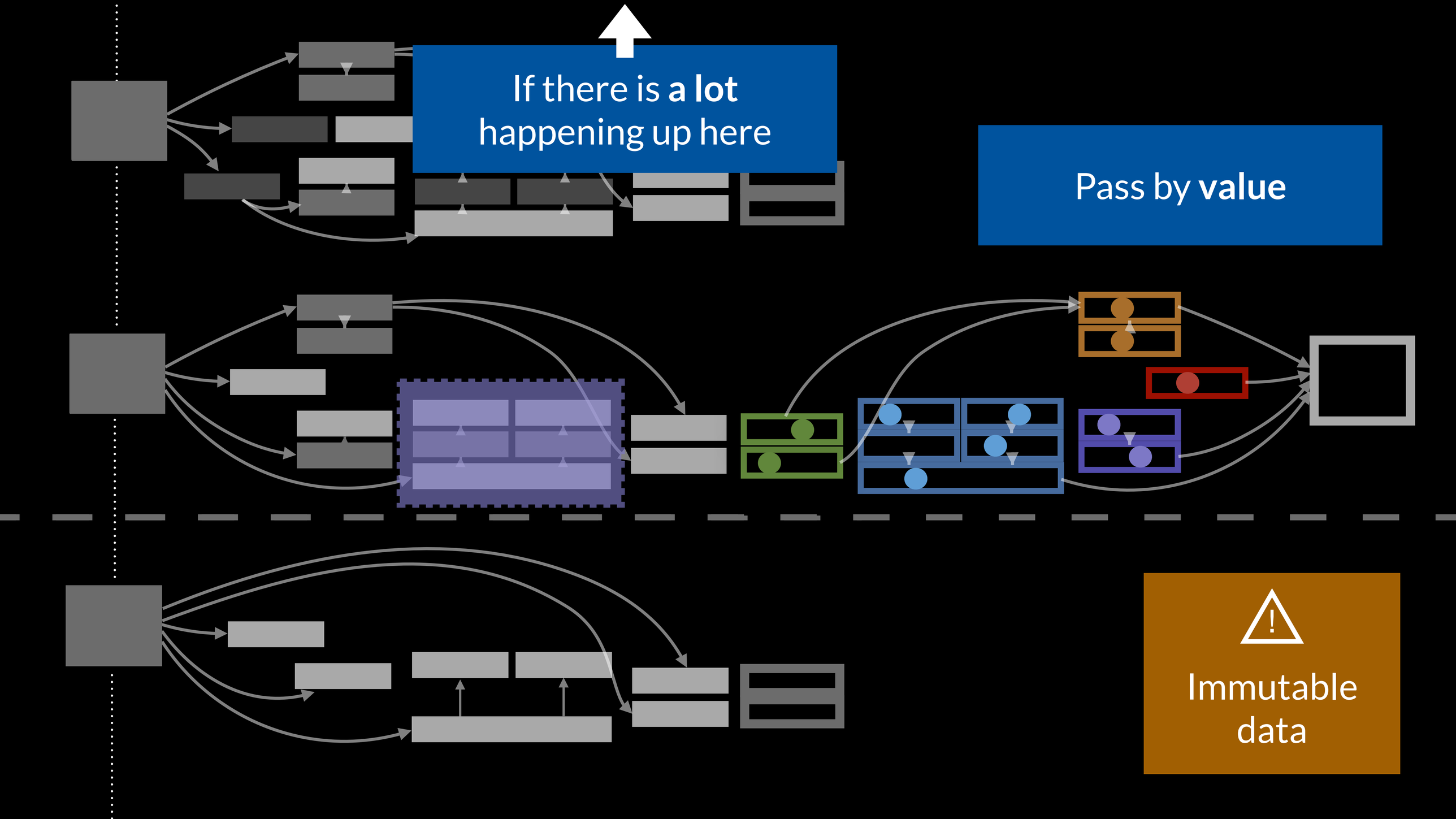


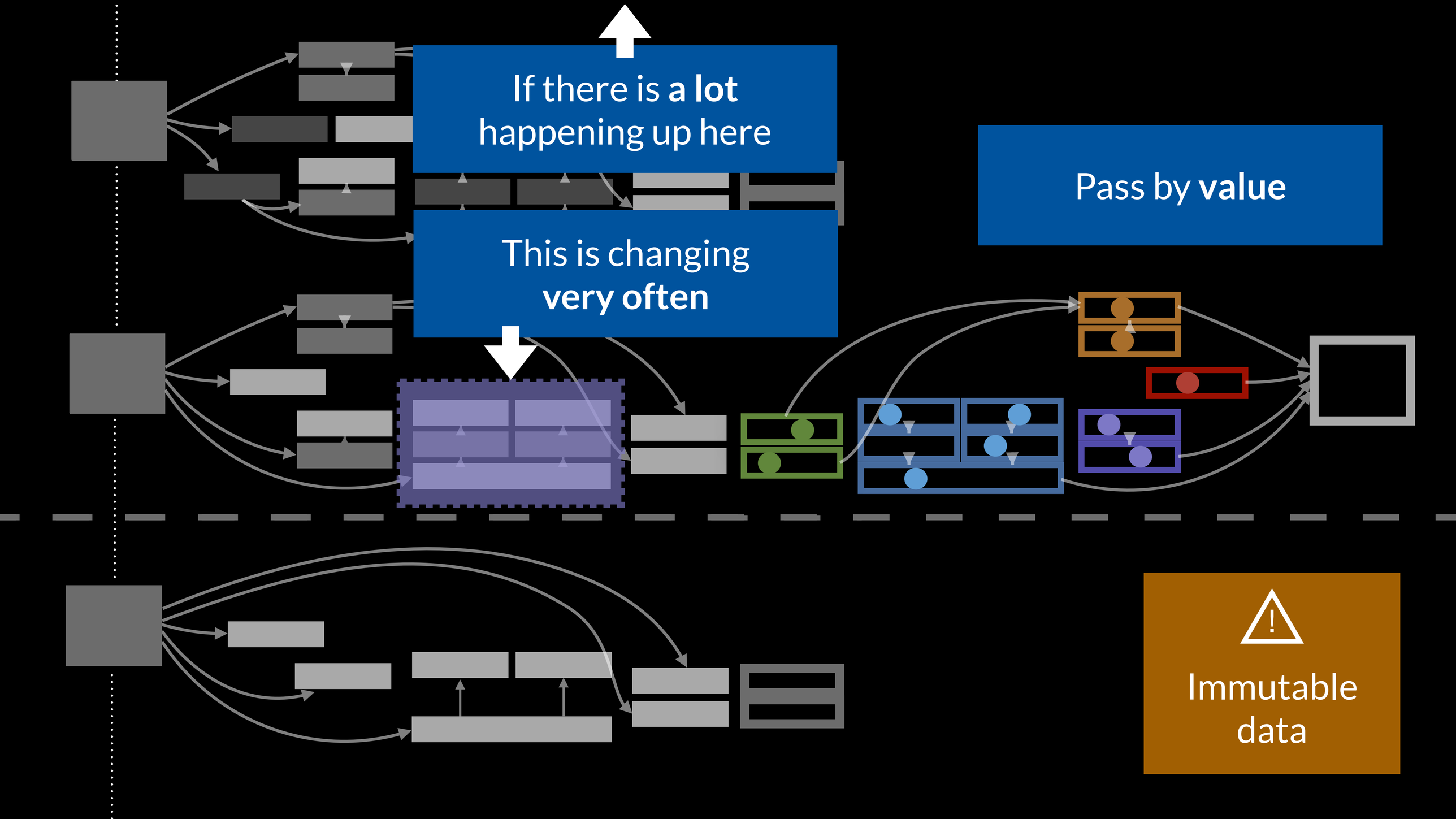

What's the catch?

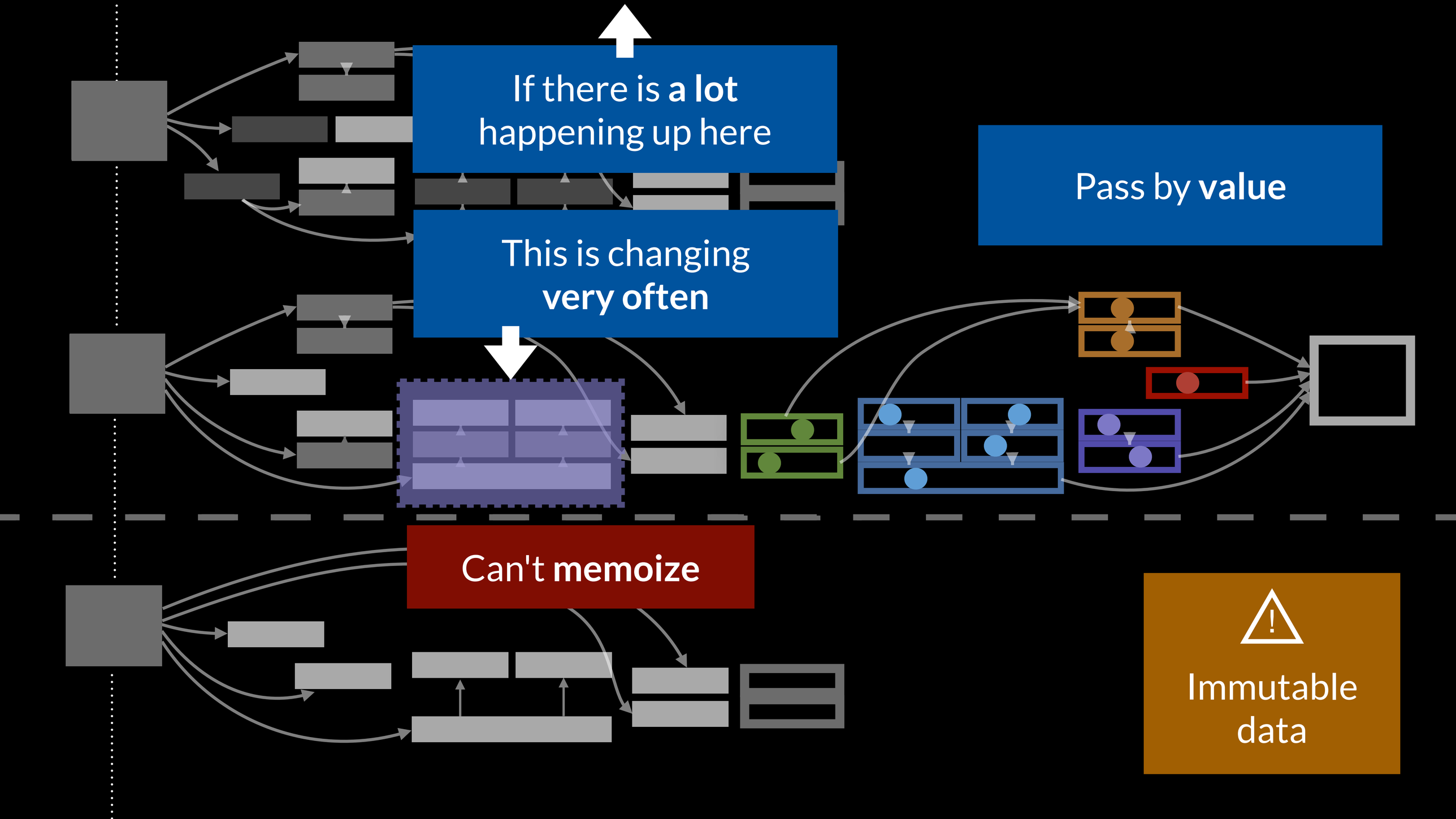

Immutable
data

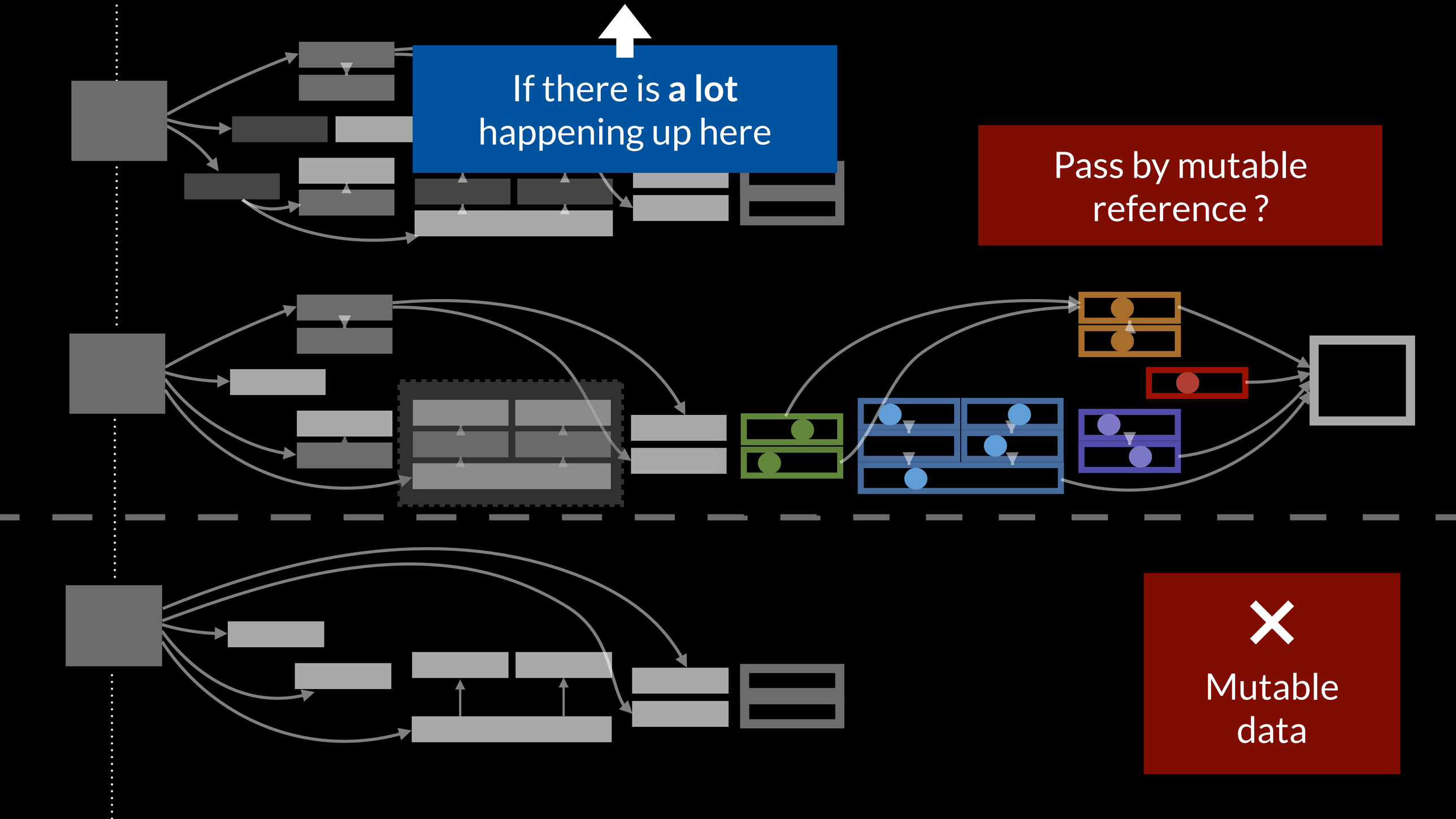

Pass by **value**
Immutable data

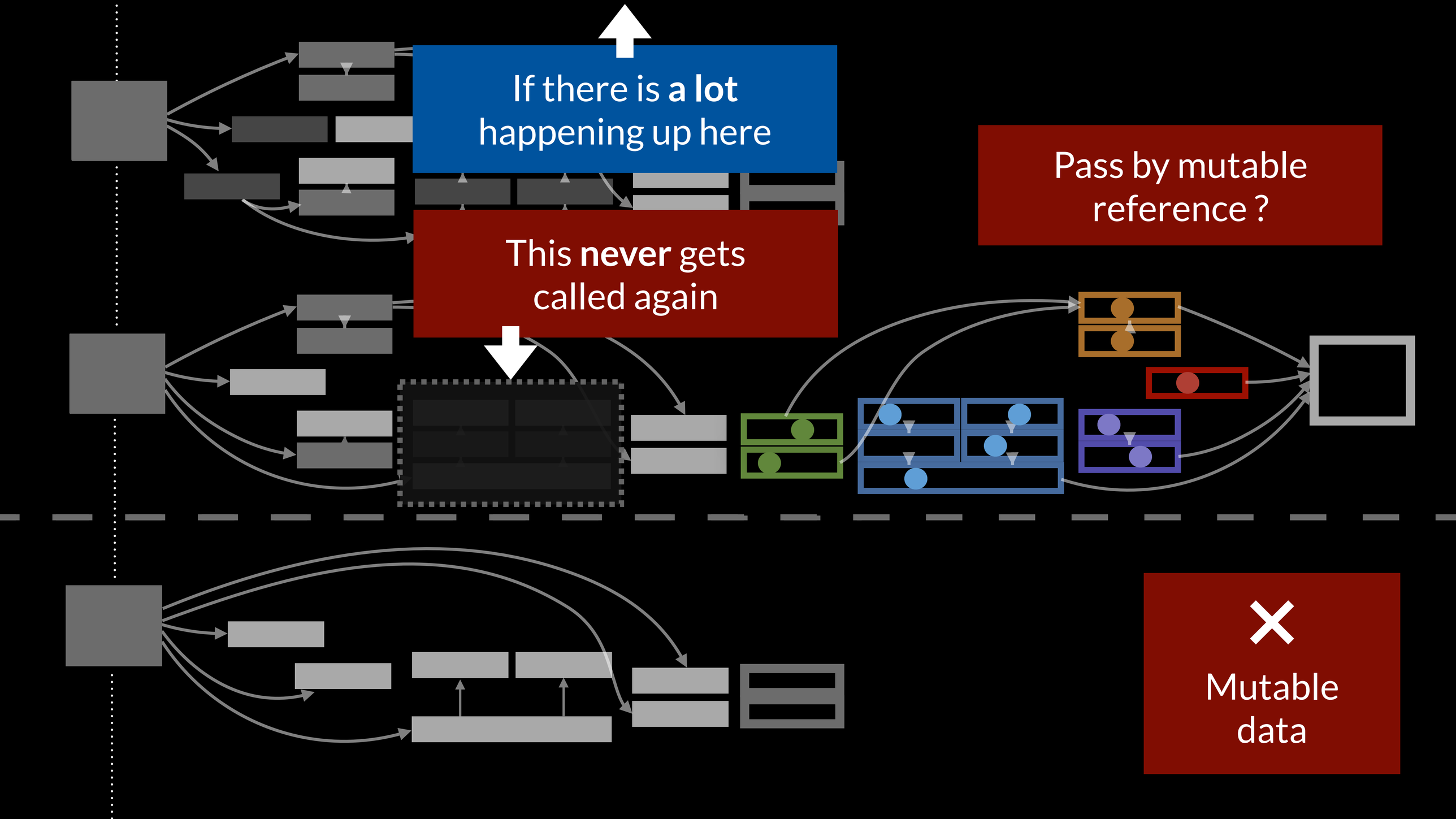

Pass by **value**
Immutable data

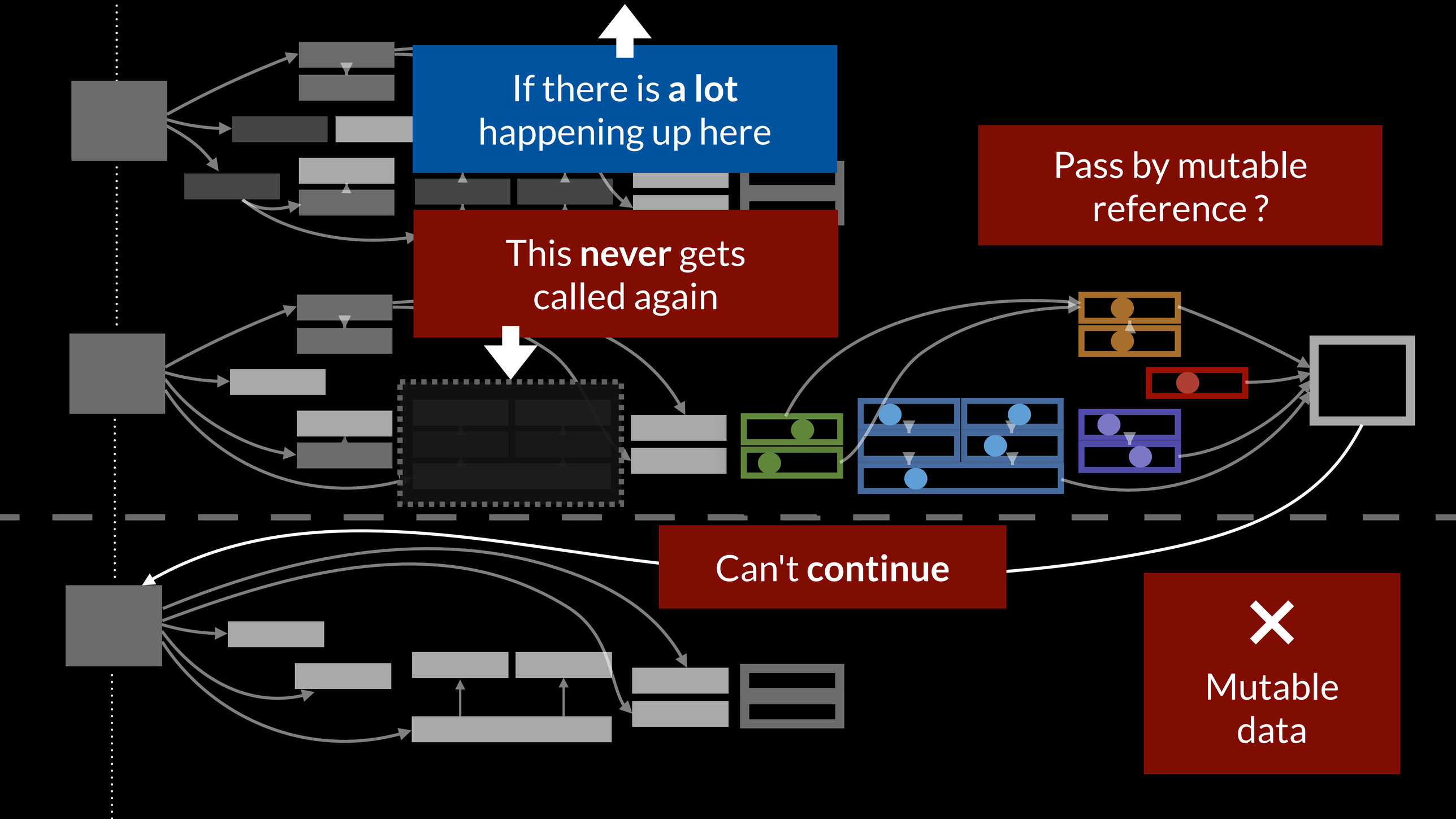

If there is **a lot** happening up here
Pass by **value**
Immutable data

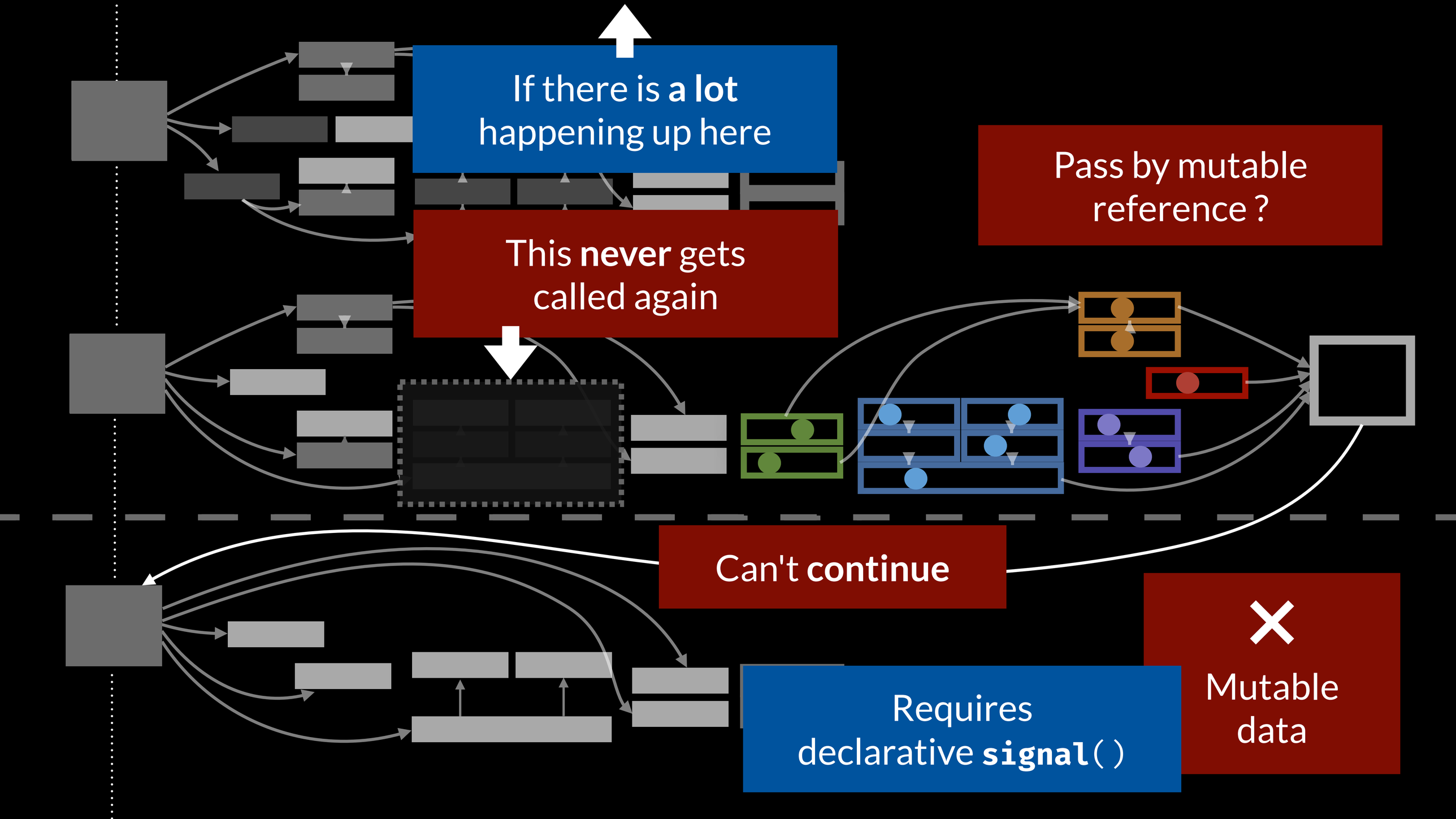

If there is **a lot** happening up here
This is changing **very often**
Pass by **value**
Immutable data

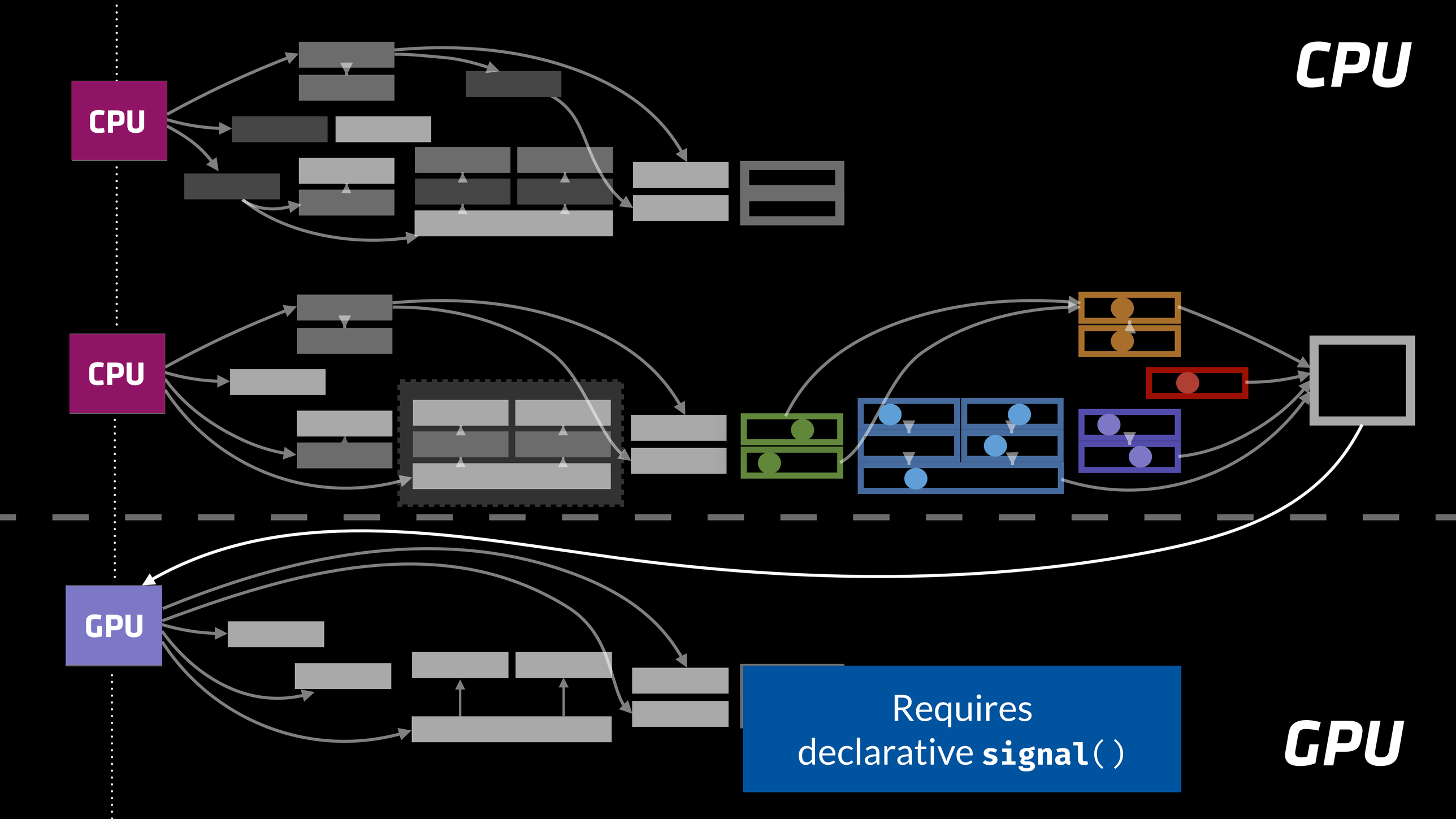

If there is **a lot** happening up here
This is changing **very often**
Pass by **value**
Can't **memoize**
Immutable data

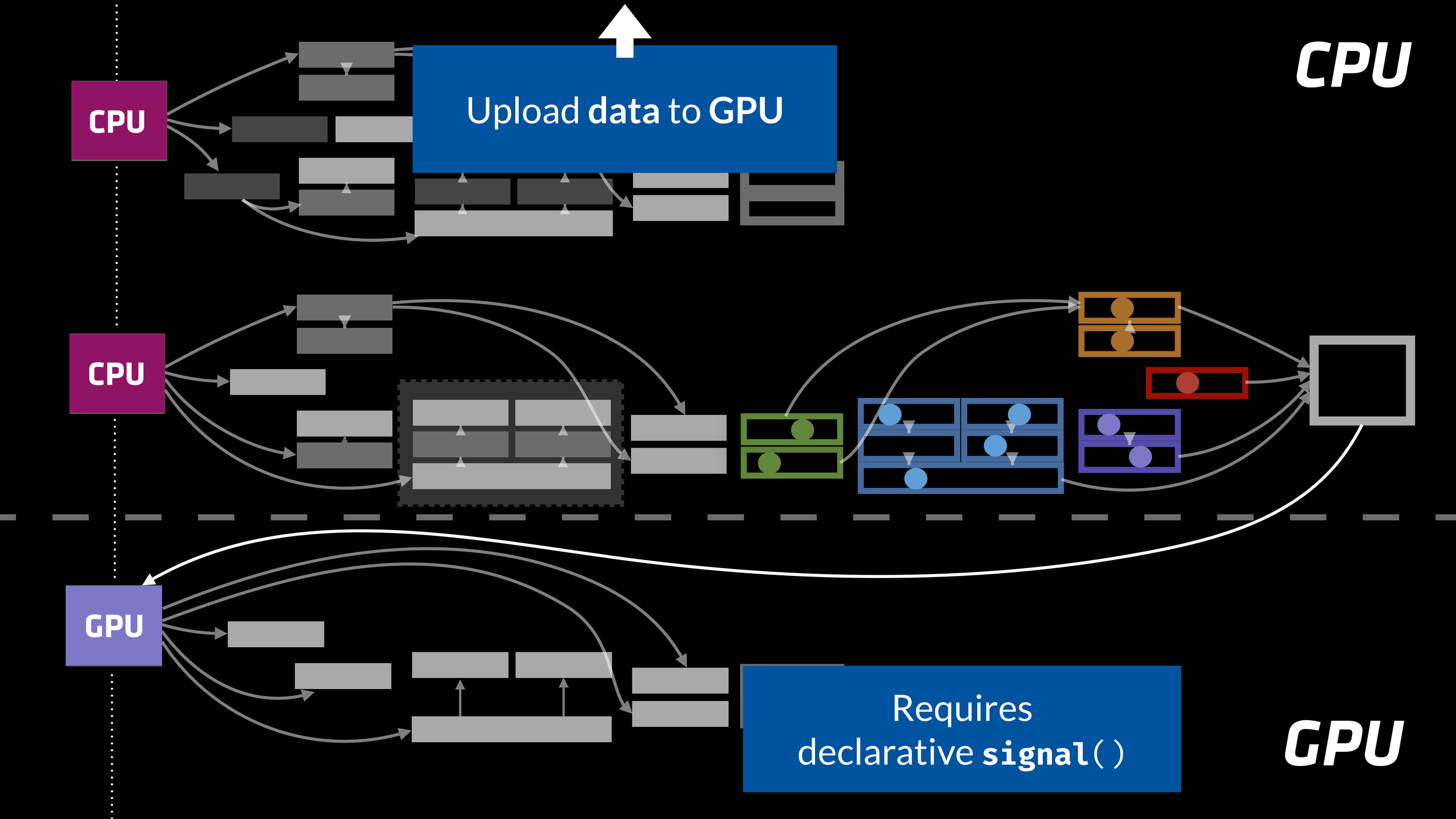

If there is **a lot** happening up here
Pass by mutable reference ?
Mutable data

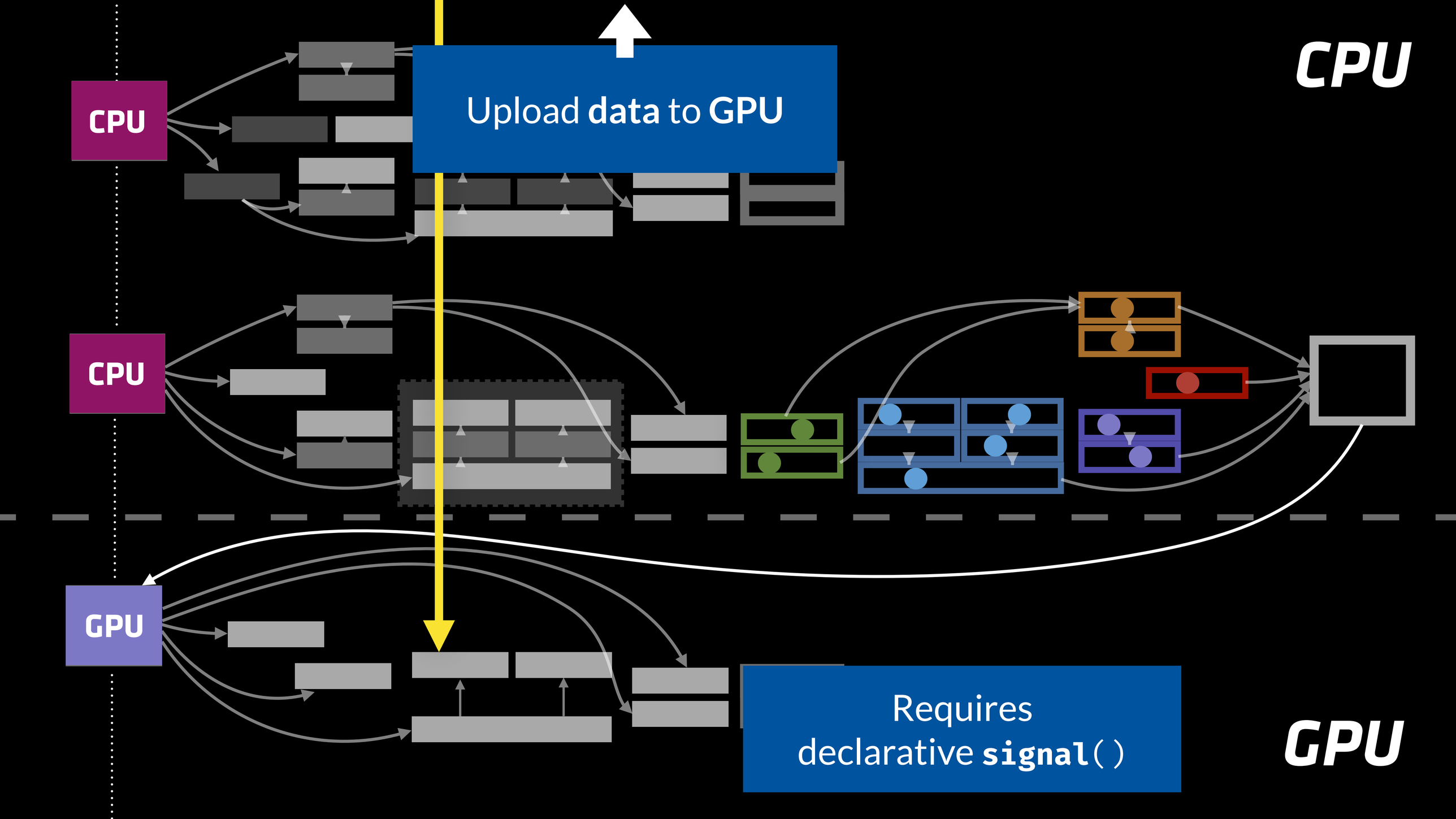

If there is **a lot** happening up here
This **never** gets called again
Pass by mutable reference ?
Mutable data

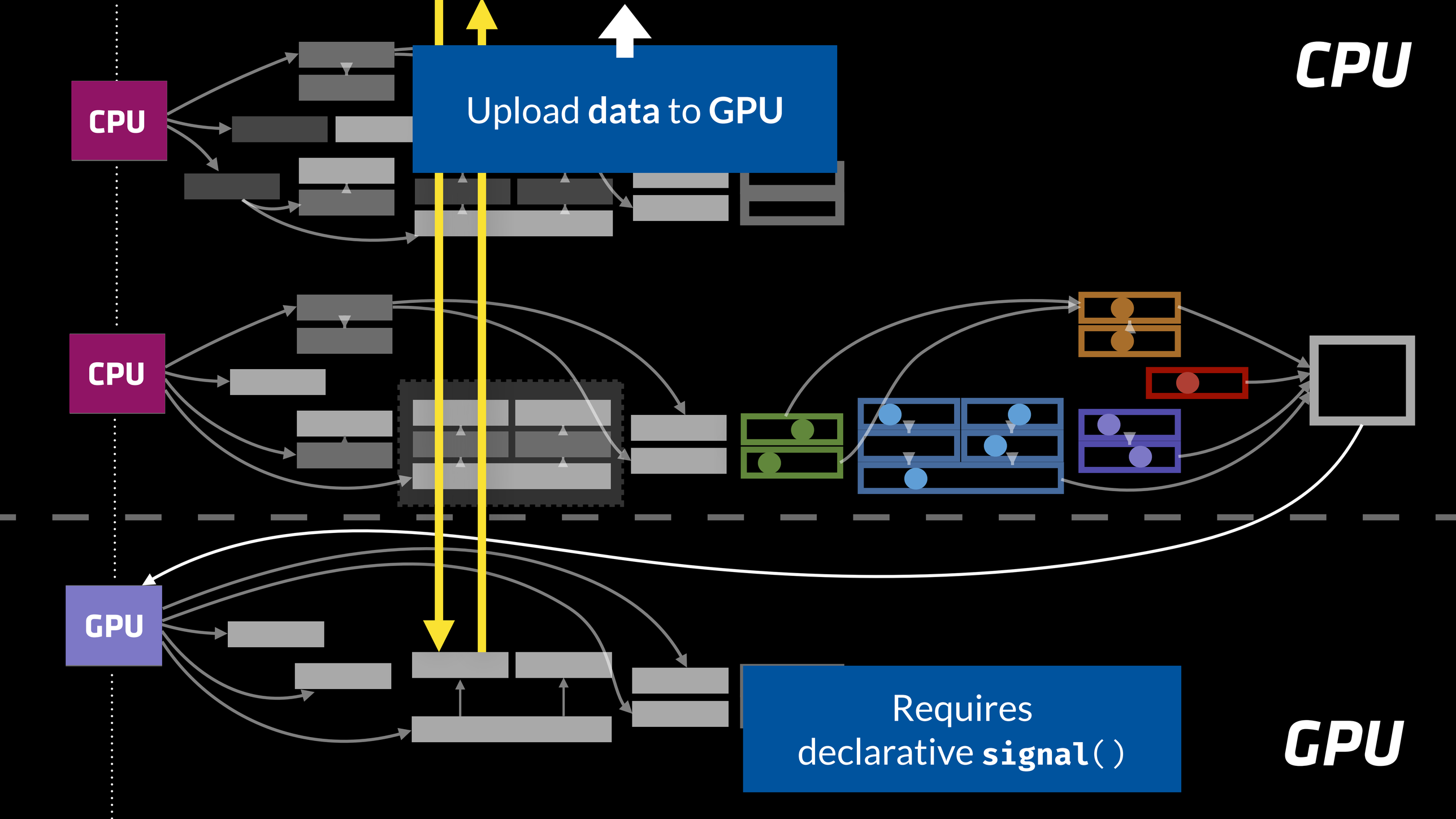

If there is **a lot** happening up here
This **never** gets called again
Pass by mutable reference ?
Can't **continue**
Mutable data

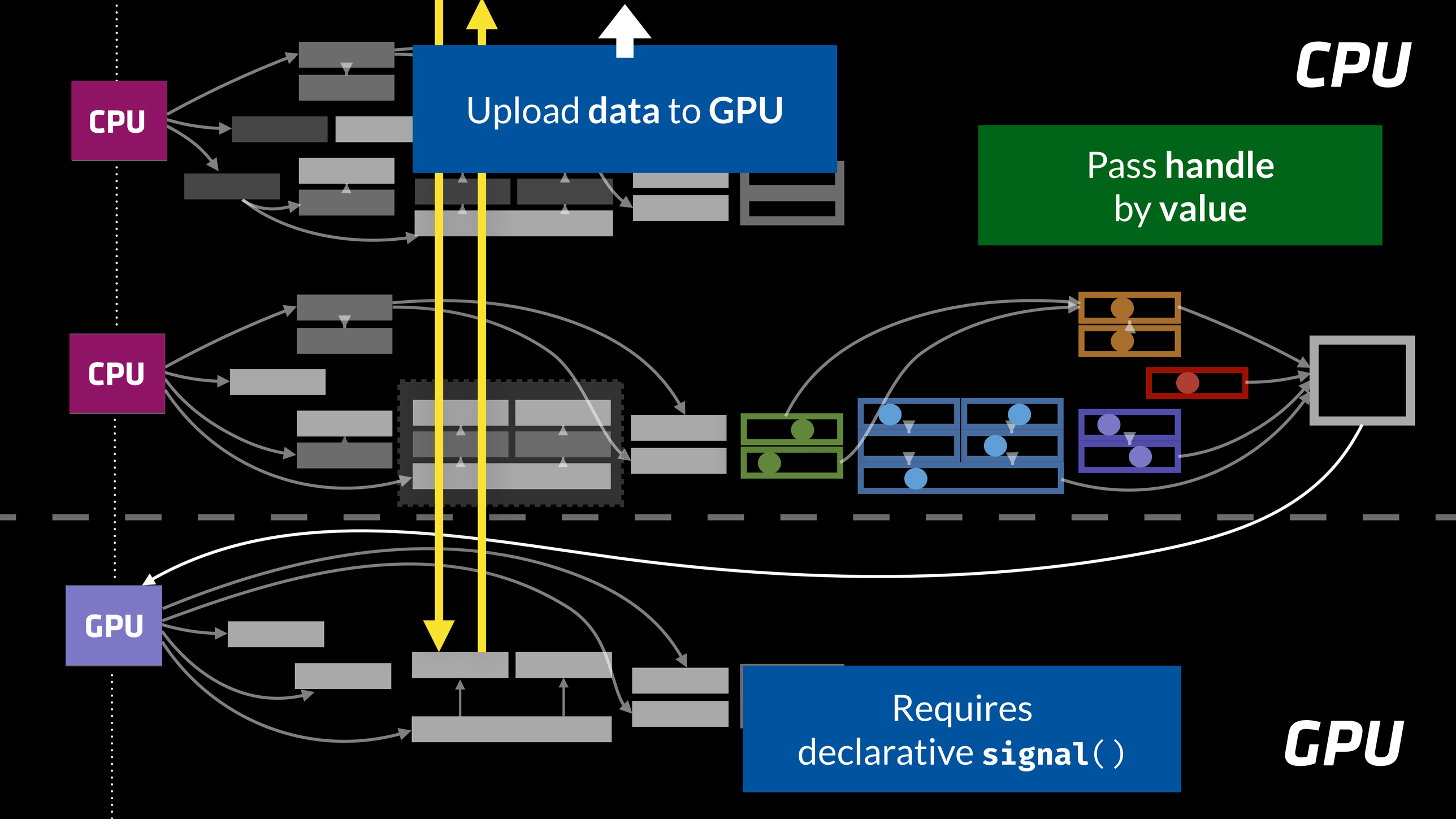

If there is **a lot** happening up here
This **never** gets called again
Pass by mutable reference ?
Can't **continue**
Requires declarative `signal()`
Mutable data

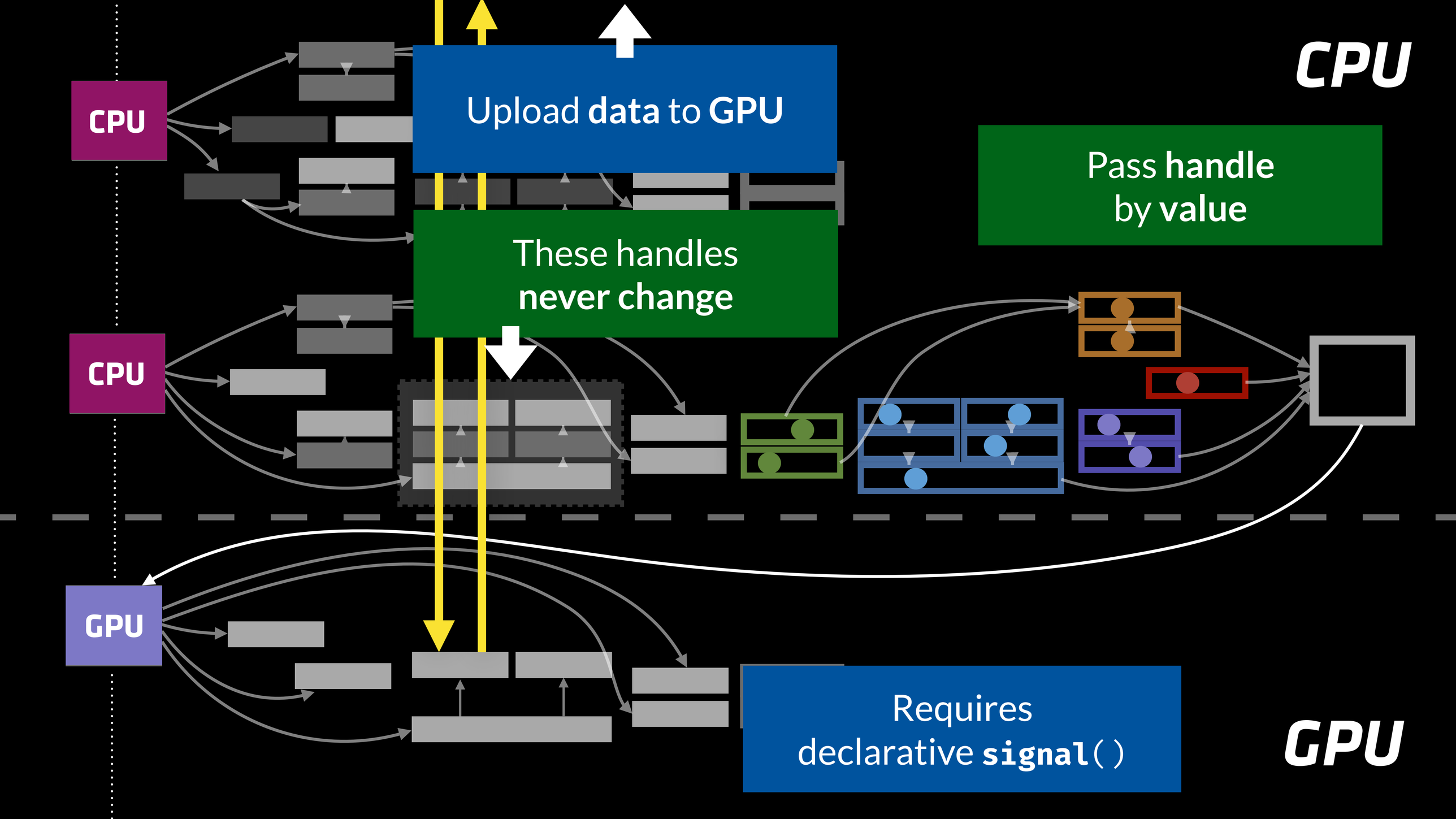

CPU
CPU
CPU
GPU
GPU
Requires
declarative signal()

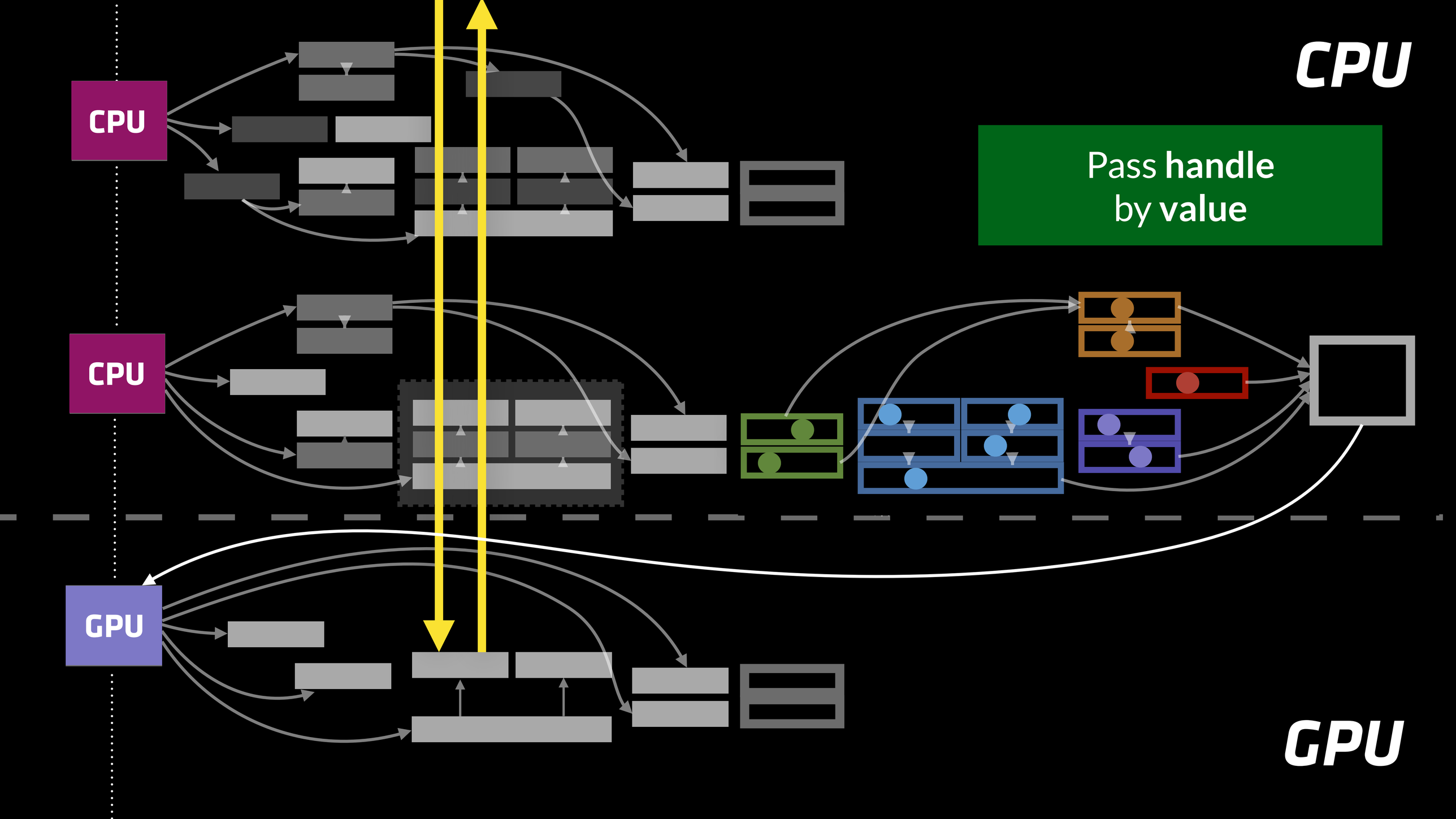

CPU
CPU
Upload **data** to **GPU**
CPU
GPU
Requires
declarative **`signal()`**
GPU

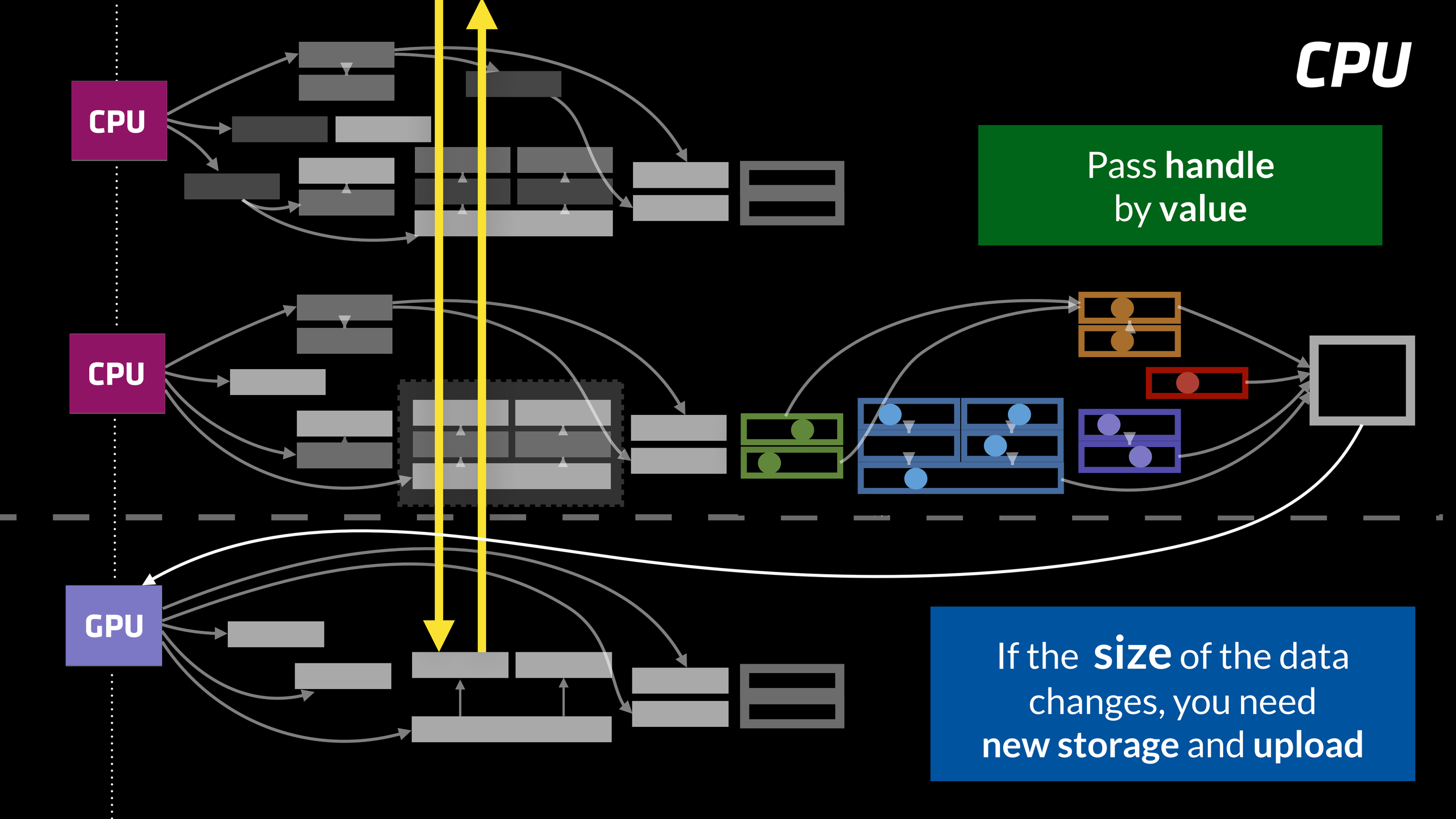

CPU
CPU
Upload **data** to **GPU**
CPU
CPU
GPU
Requires
declarative **`signal`**`()`
GPU

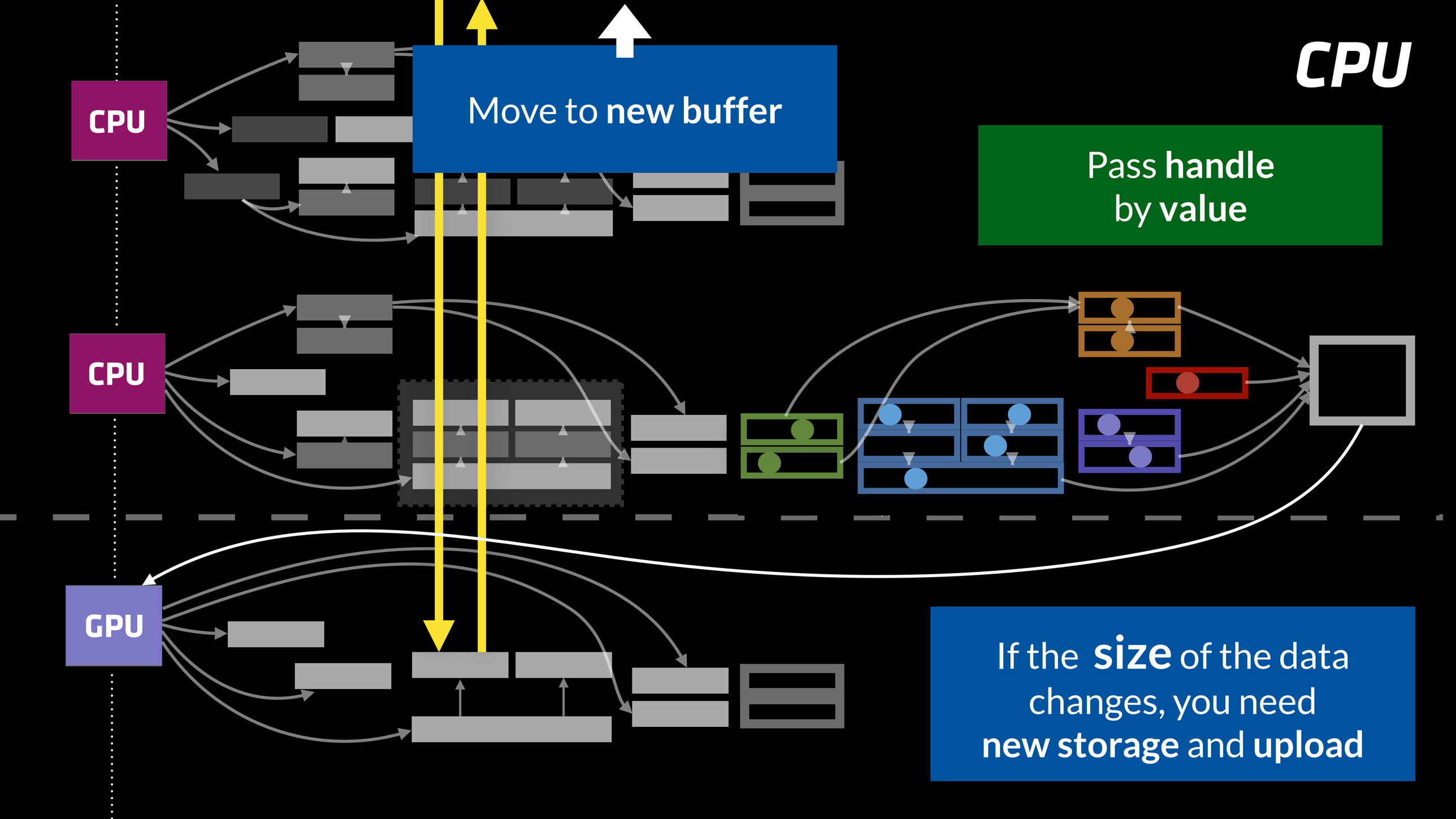

CPU
CPU
Upload **data** to **GPU**
CPU
GPU
Requires
declarative **signal**()
GPU

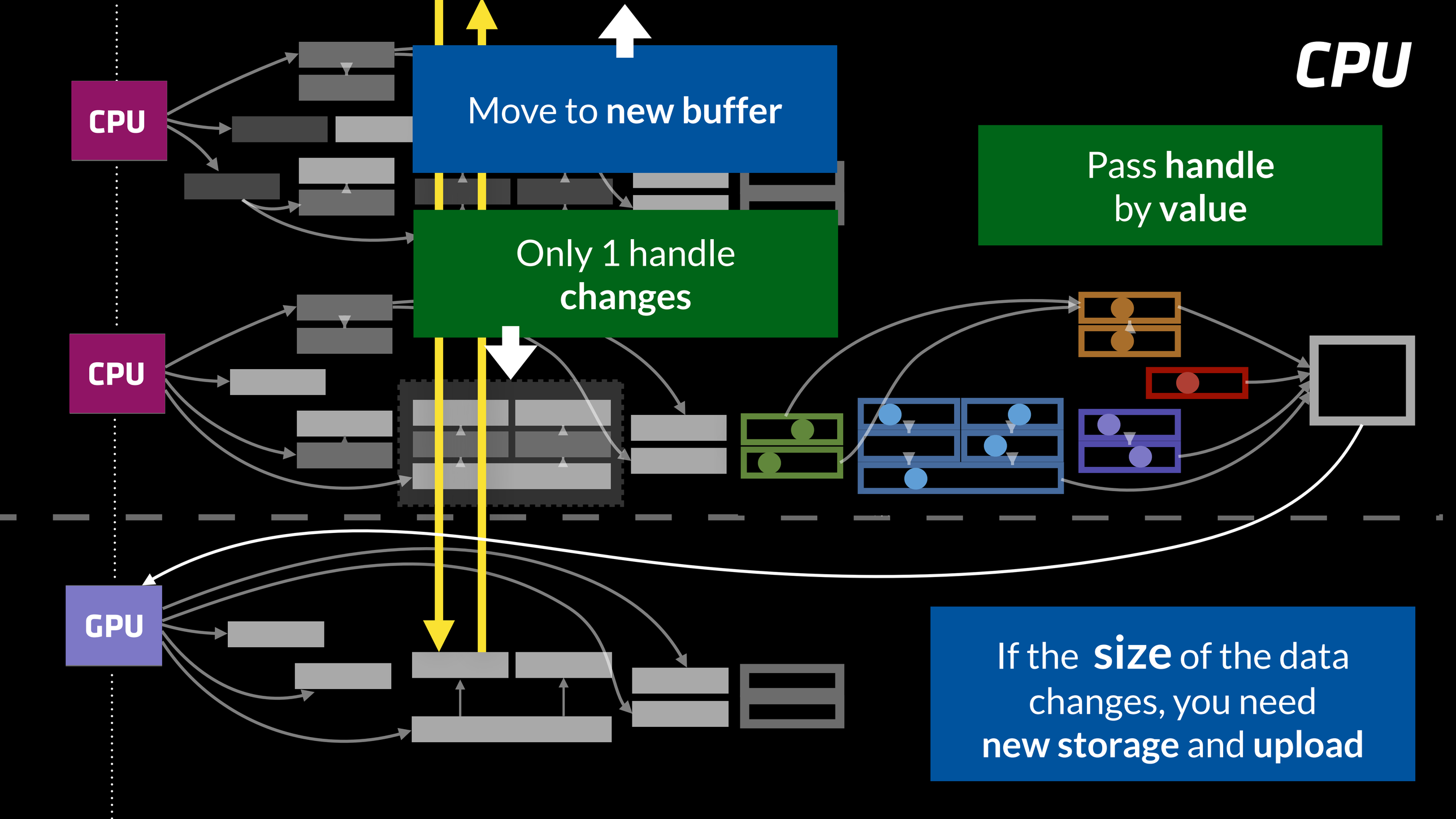

CPU
CPU
Upload **data** to **GPU**
Pass **handle**
by **value**
CPU
GPU
Requires
declarative **`signal()`**
GPU

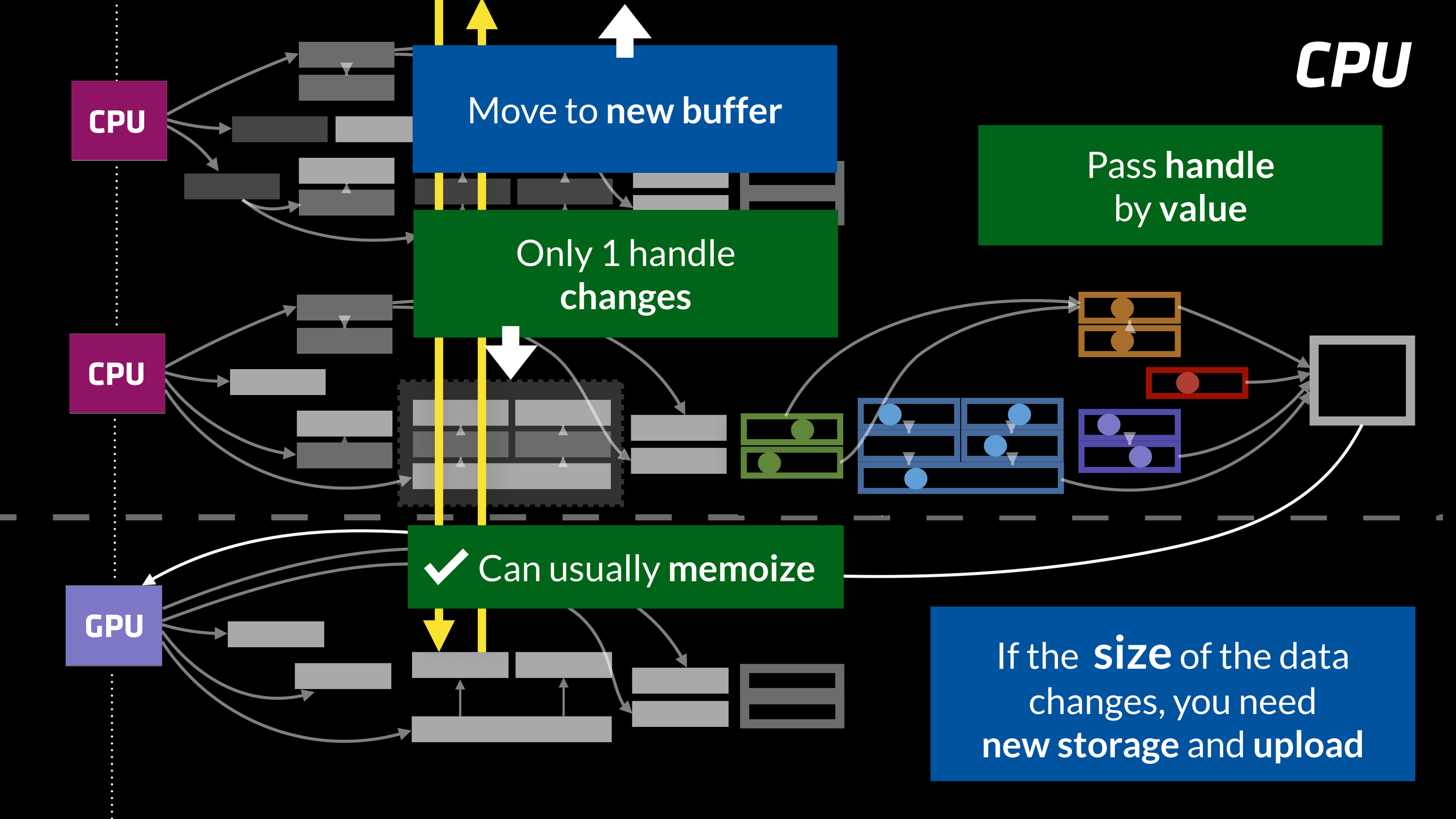

CPU
CPU
Upload **data** to **GPU**
Pass **handle**
by **value**
These handles
never change
CPU
GPU
Requires
declarative **`signal()`**
GPU

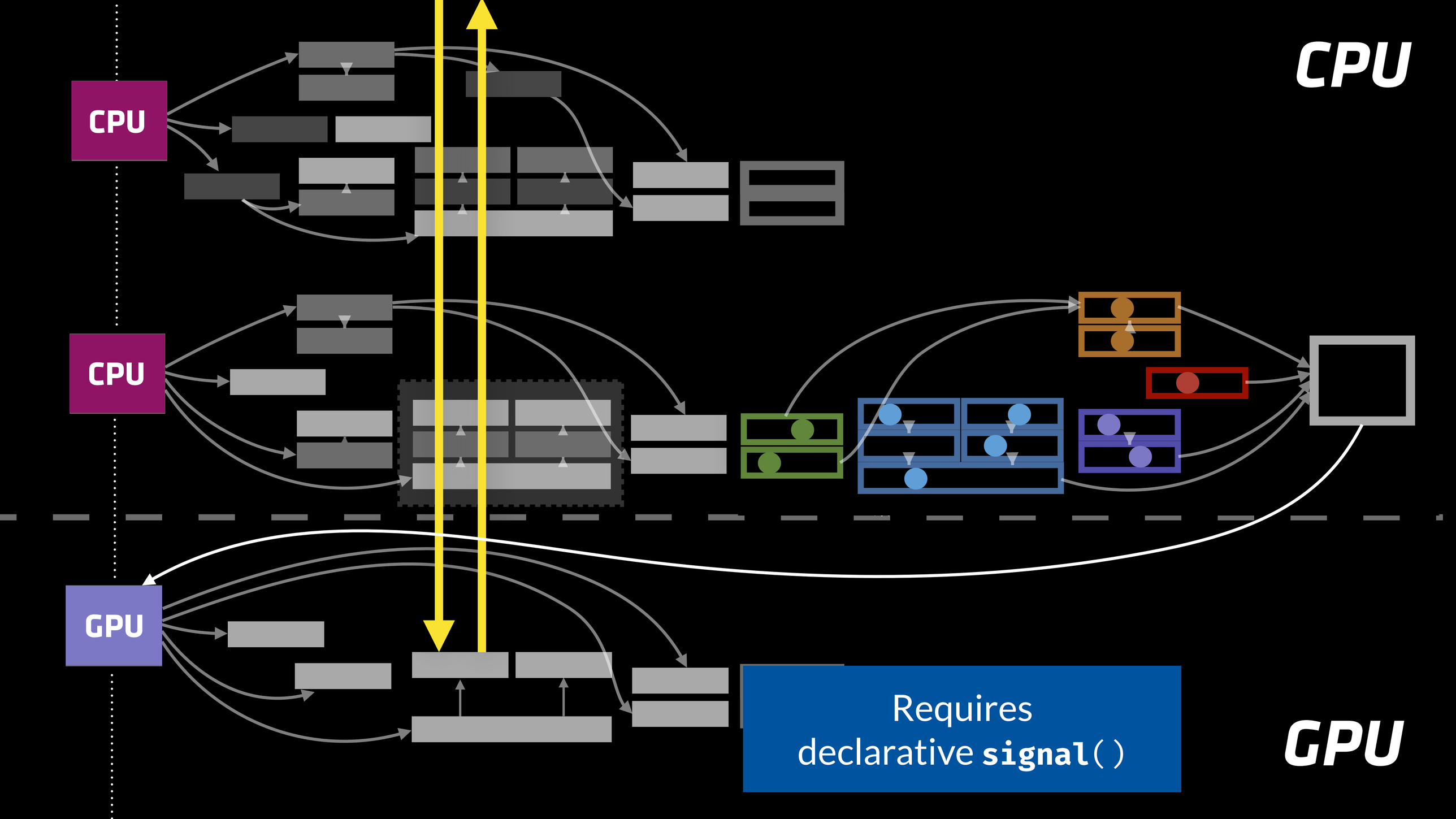

CPU
CPU
CPU
Pass handle
by value
GPU
GPU

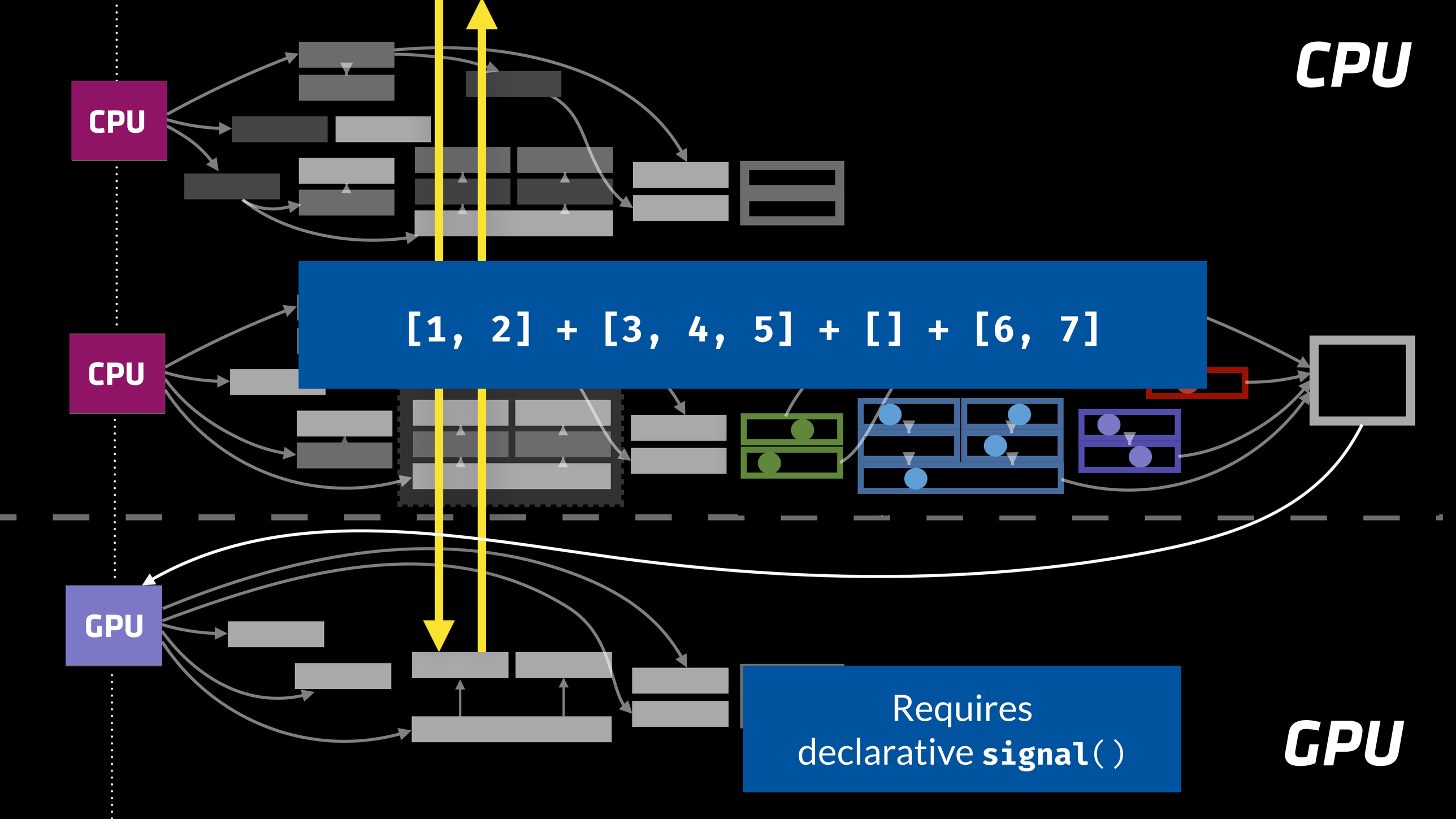

CPU
CPU
CPU
CPU
Pass **handle**
by **value**
GPU
If the **size** of the data
changes, you need
new storage and **upload**

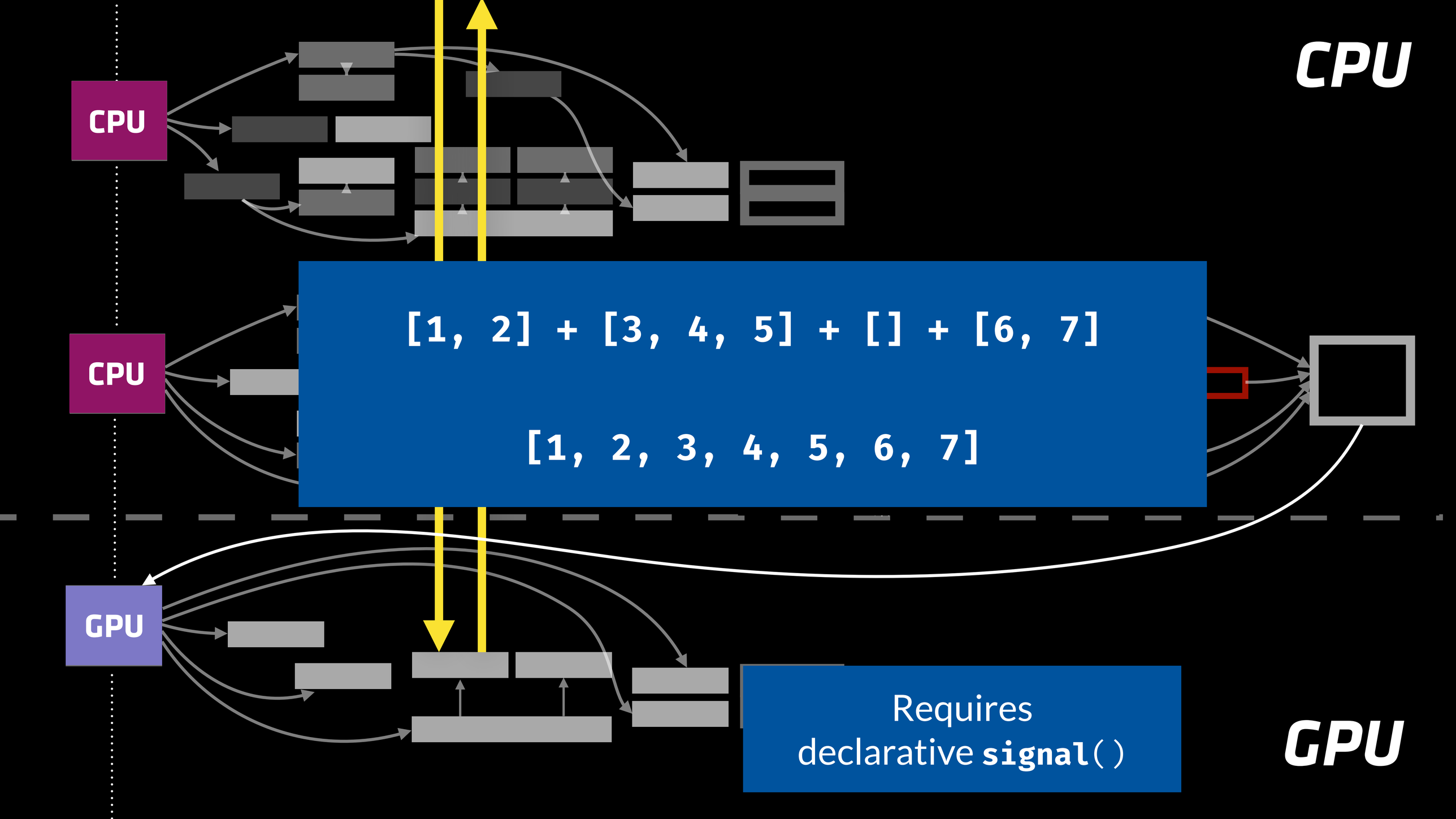

CPU
CPU
CPU
GPU
Move to **new buffer**
Pass **handle** by **value**
If the **size** of the data changes, you need **new storage** and **upload**

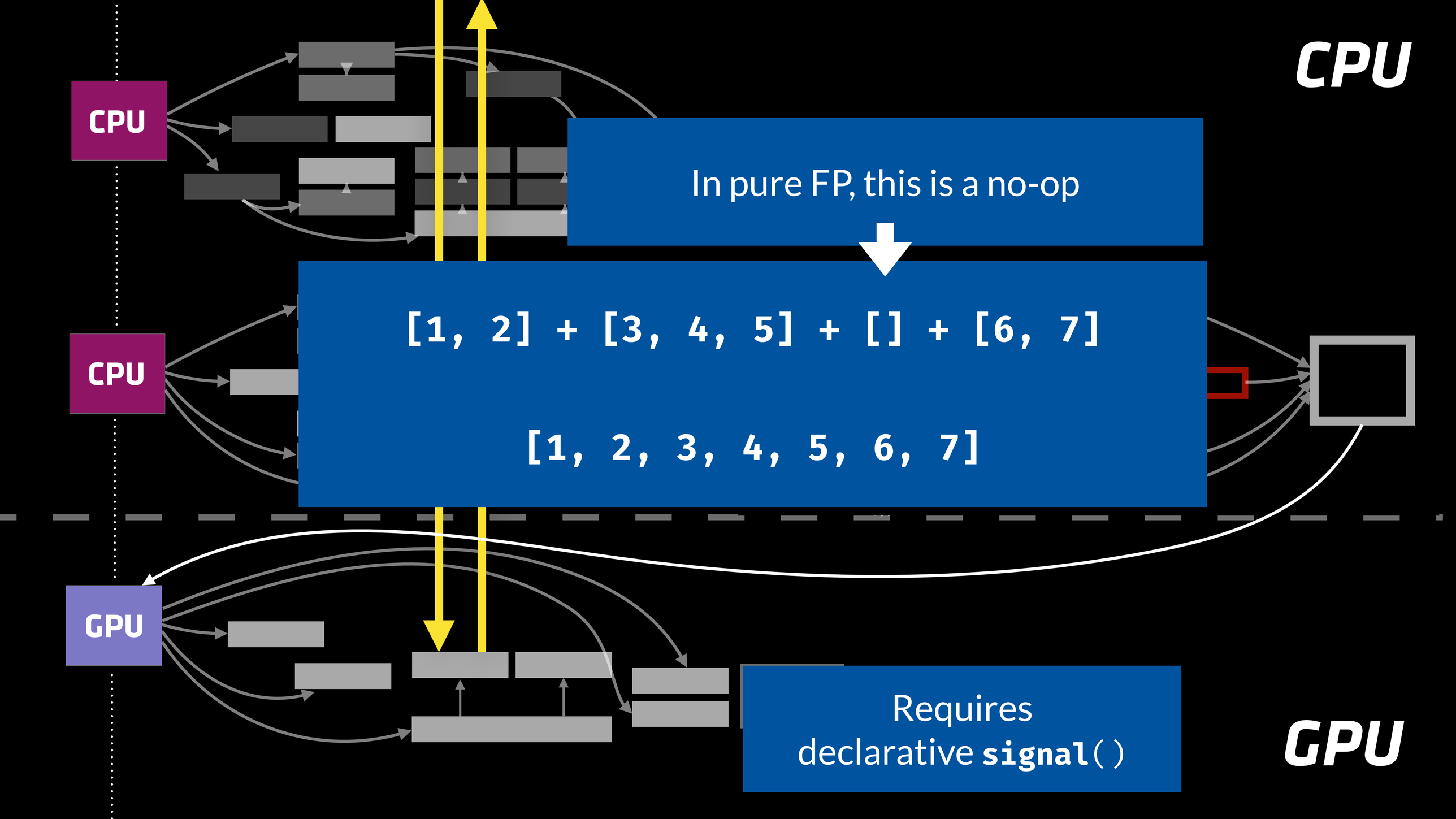

CPU
CPU
CPU
CPU
Move to **new buffer**
Only 1 handle
changes
Pass **handle**
by **value**
GPU
If the **size** of the data
changes, you need
new storage and **upload**

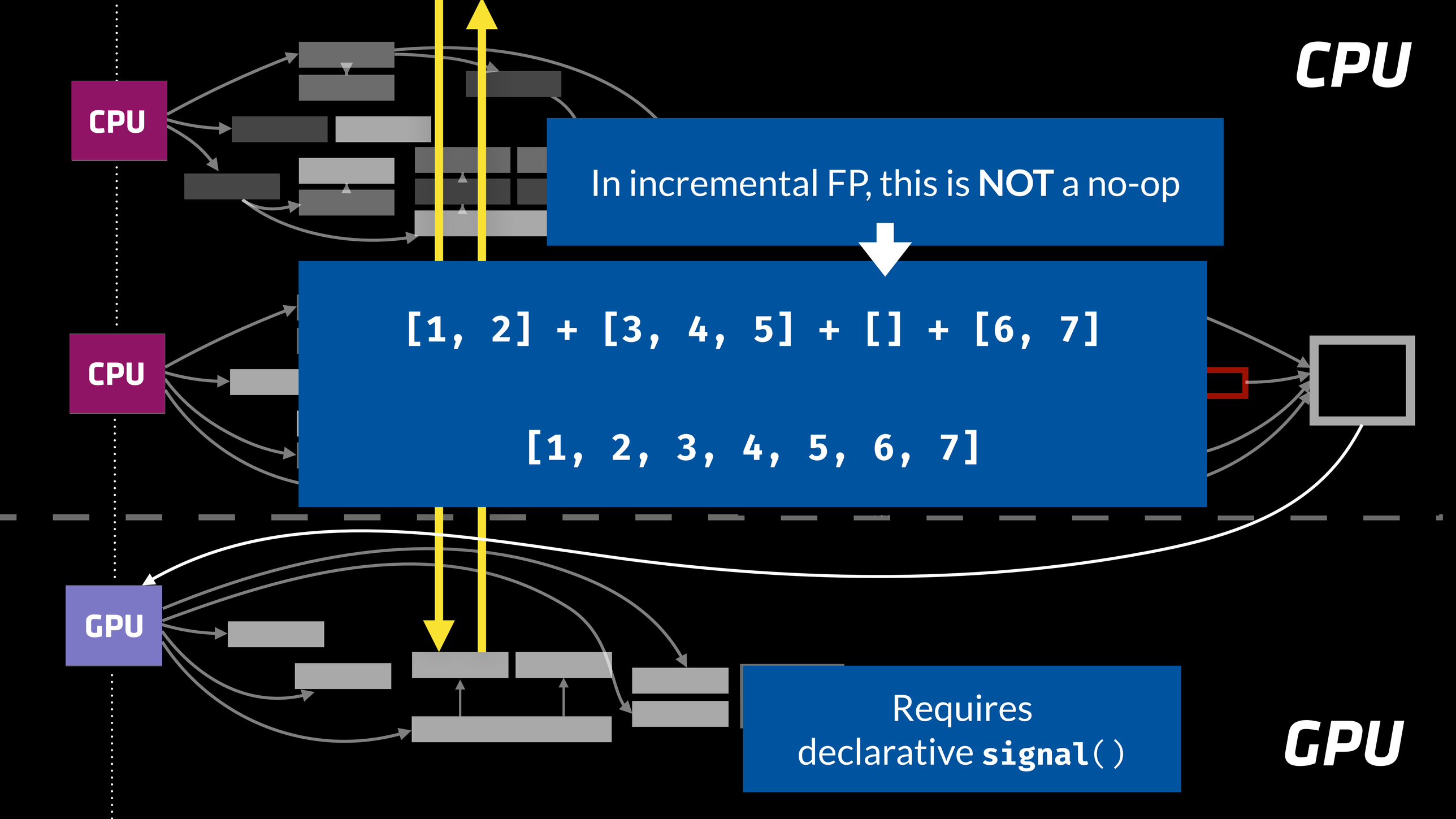

CPU
CPU
CPU
CPU
Move to **new buffer**
Only 1 handle **changes**
Pass **handle** by **value**
✓ Can usually **memoize**
GPU
If the **size** of the data changes, you need **new storage** and **upload**

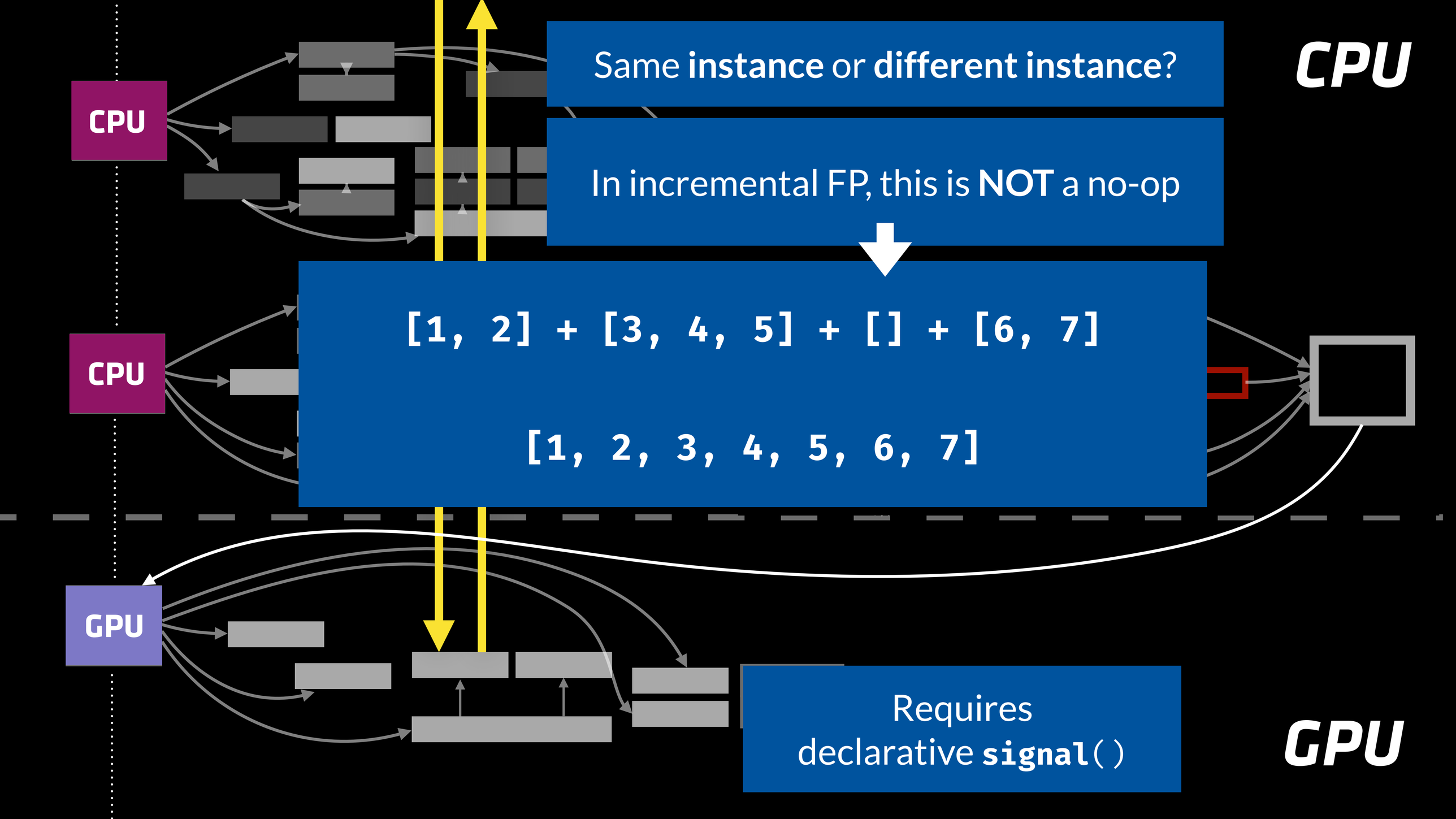

CPU
CPU
CPU
GPU
GPU
Requires
declarative signal()

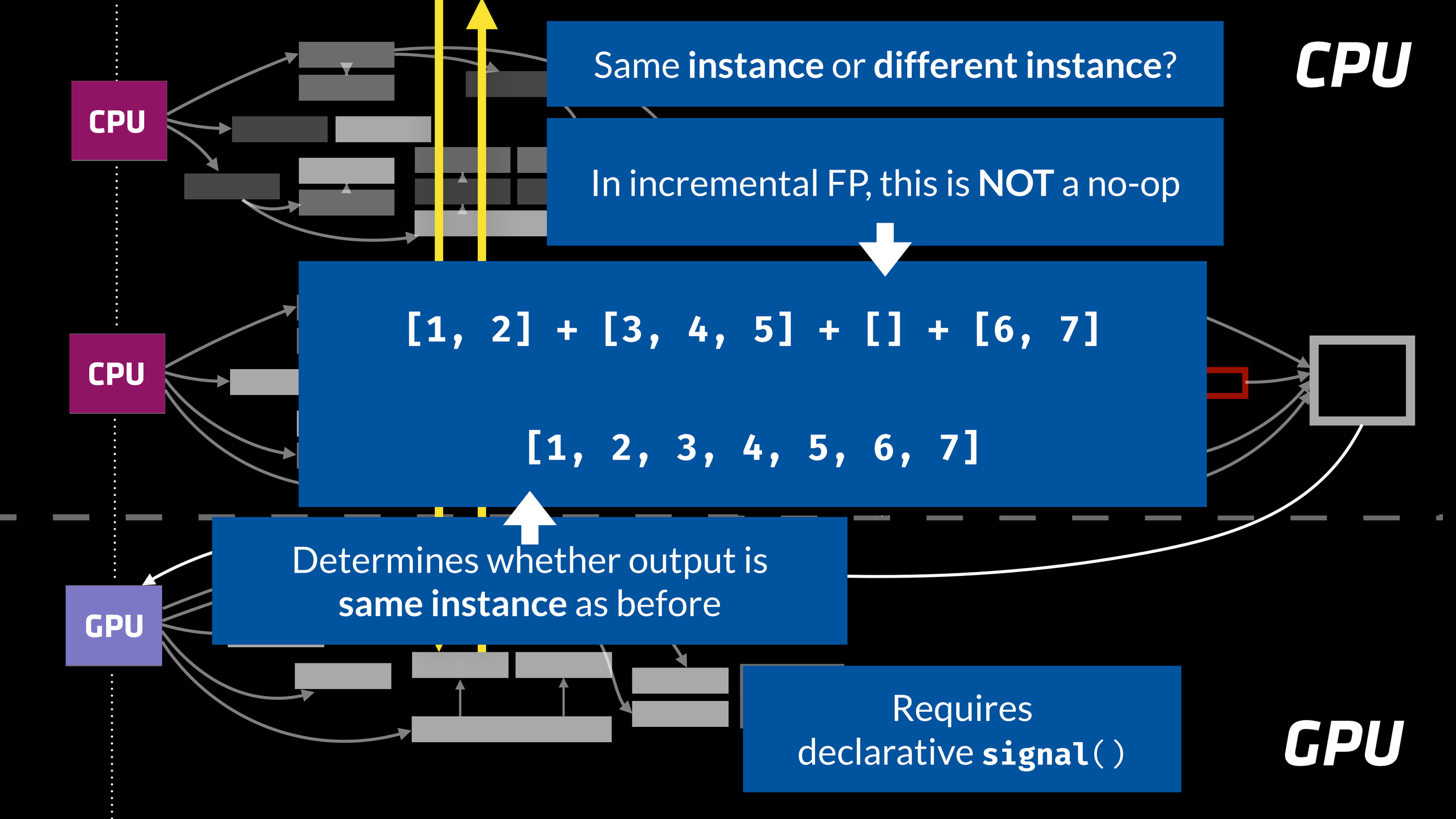

CPU
CPU
CPU
[1, 2] + [3, 4, 5] + [] + [6, 7]
GPU
Requires
declarative signal()
GPU

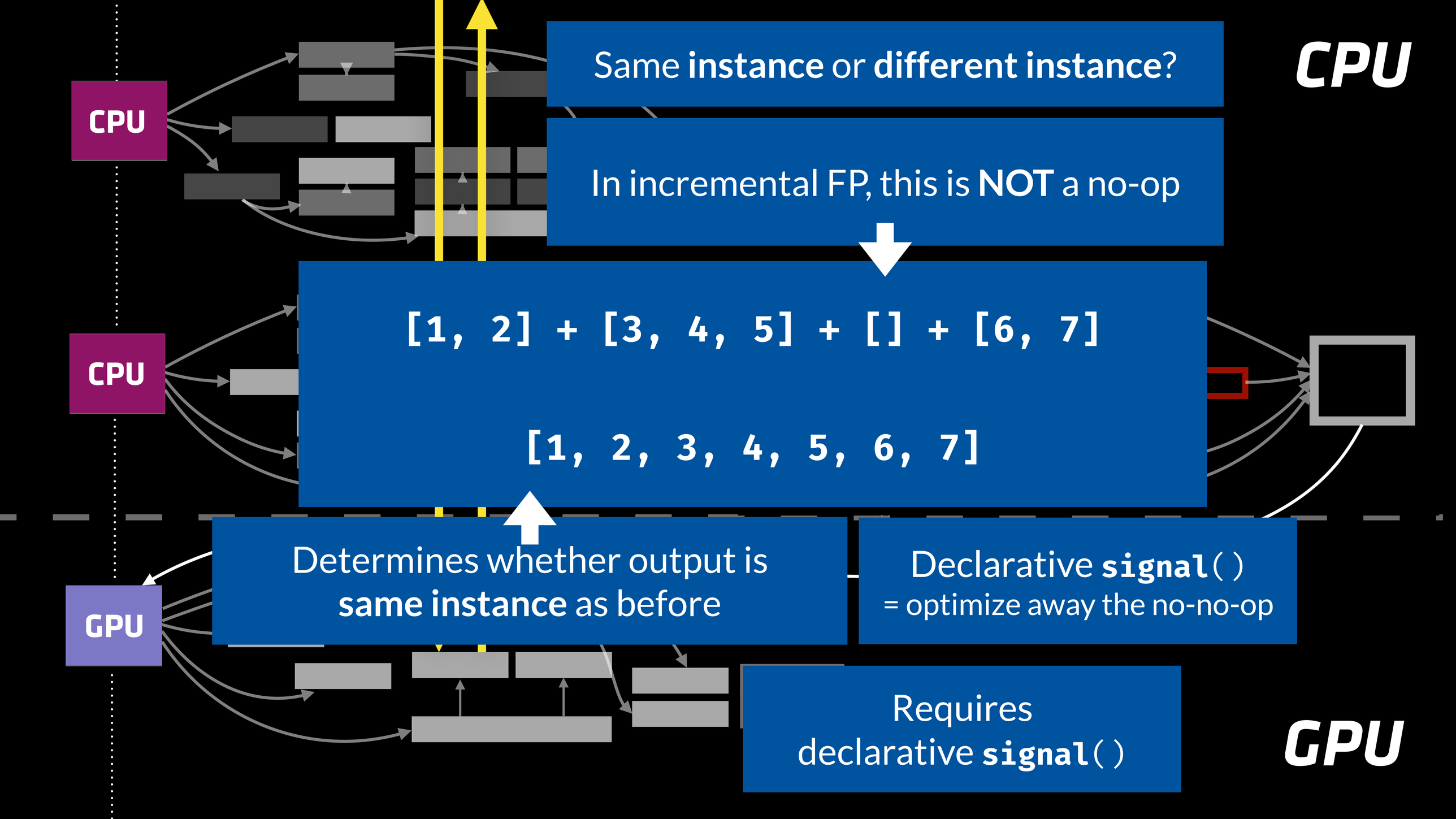

CPU
CPU
CPU
[1, 2] + [3, 4, 5] + [] + [6, 7]
[1, 2, 3, 4, 5, 6, 7]
GPU
Requires
declarative signal()
GPU

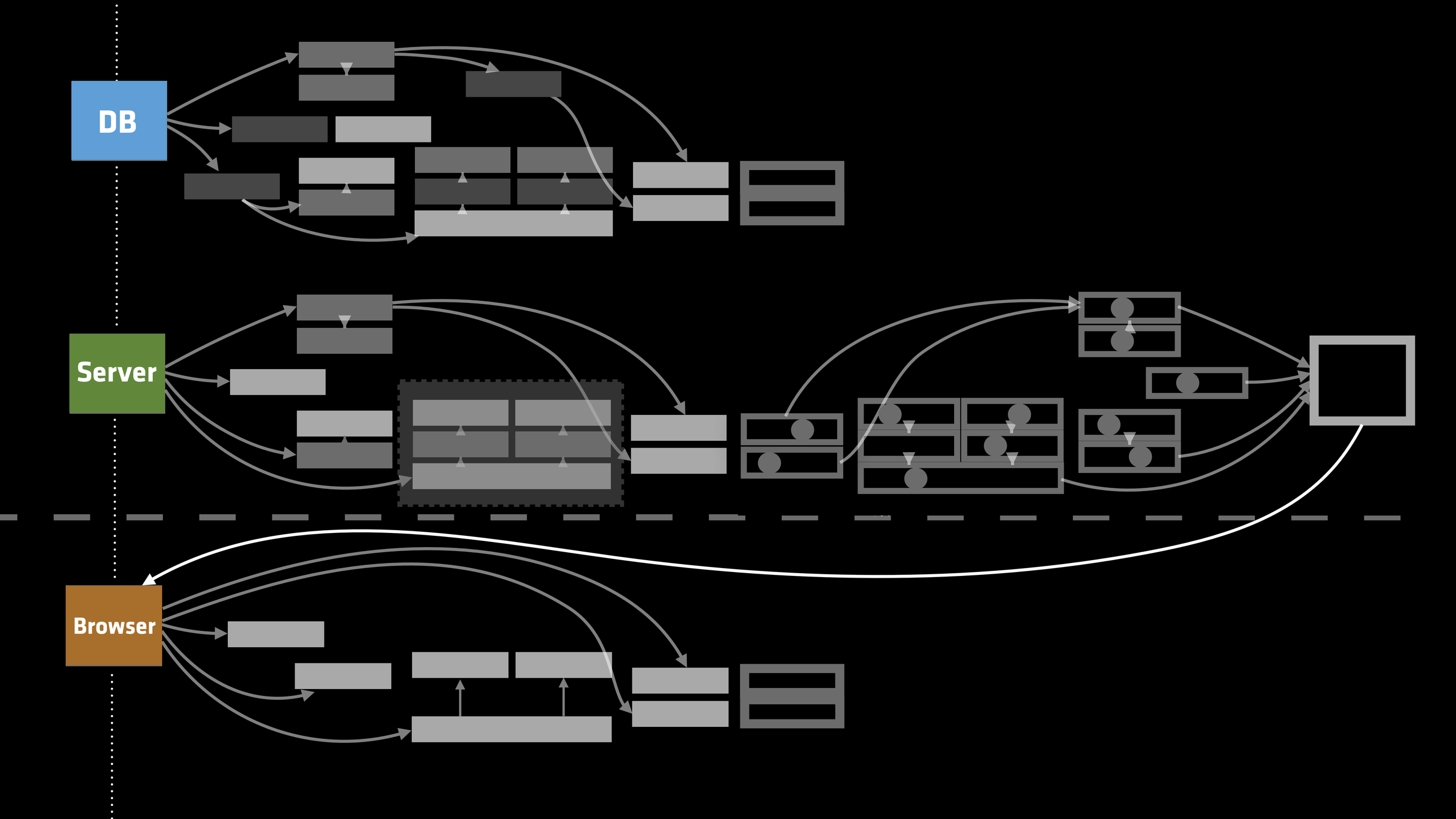

CPU
CPU
CPU
In pure FP, this is a no-op
[1, 2] + [3, 4, 5] + [] + [6, 7]
[1, 2, 3, 4, 5, 6, 7]
GPU
Requires
declarative signal()
GPU

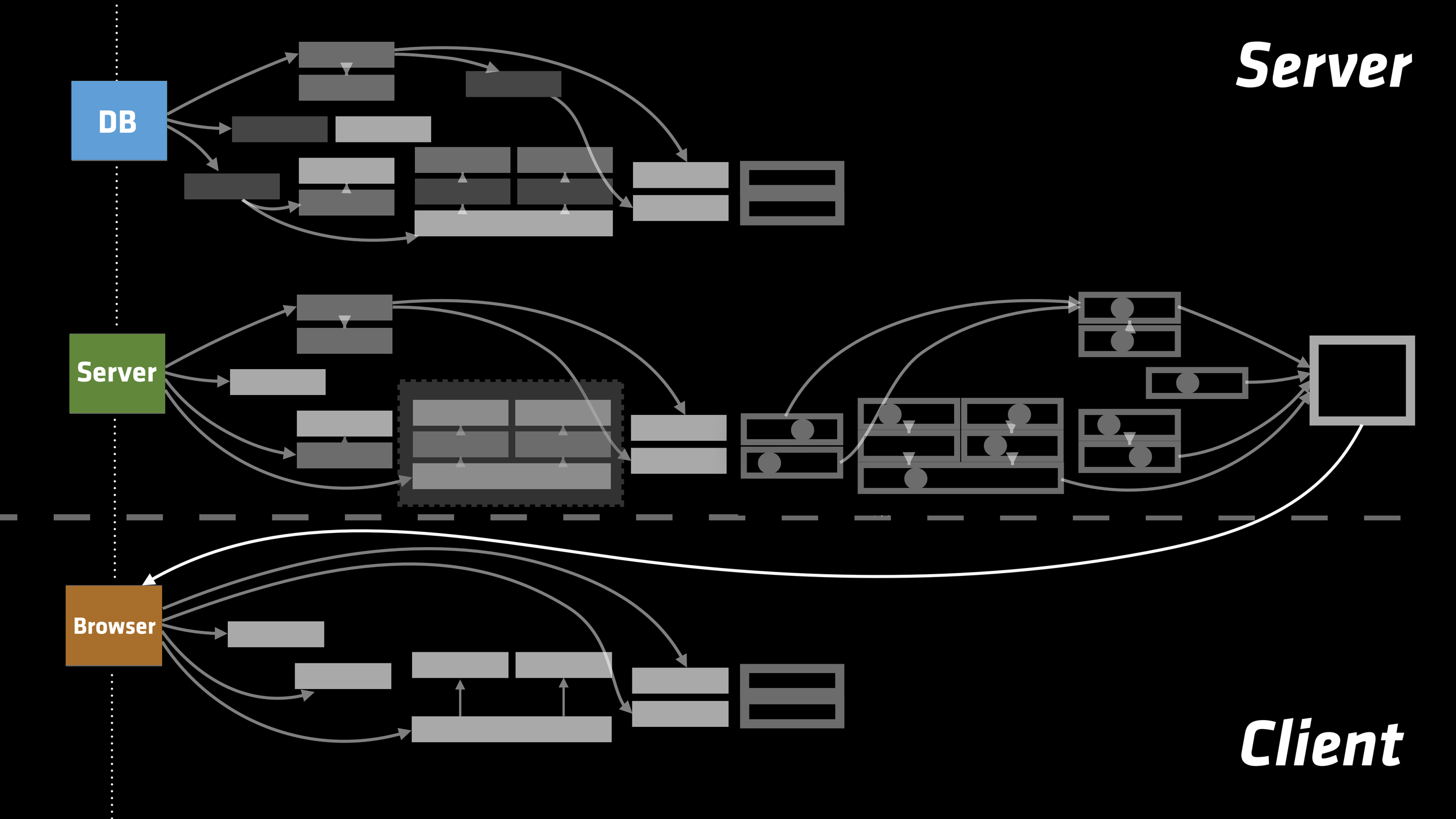

CPU
CPU
CPU
In incremental FP, this is **NOT** a no-op
[1, 2] + [3, 4, 5] + [] + [6, 7]
[1, 2, 3, 4, 5, 6, 7]
GPU
Requires
declarative signal()
GPU

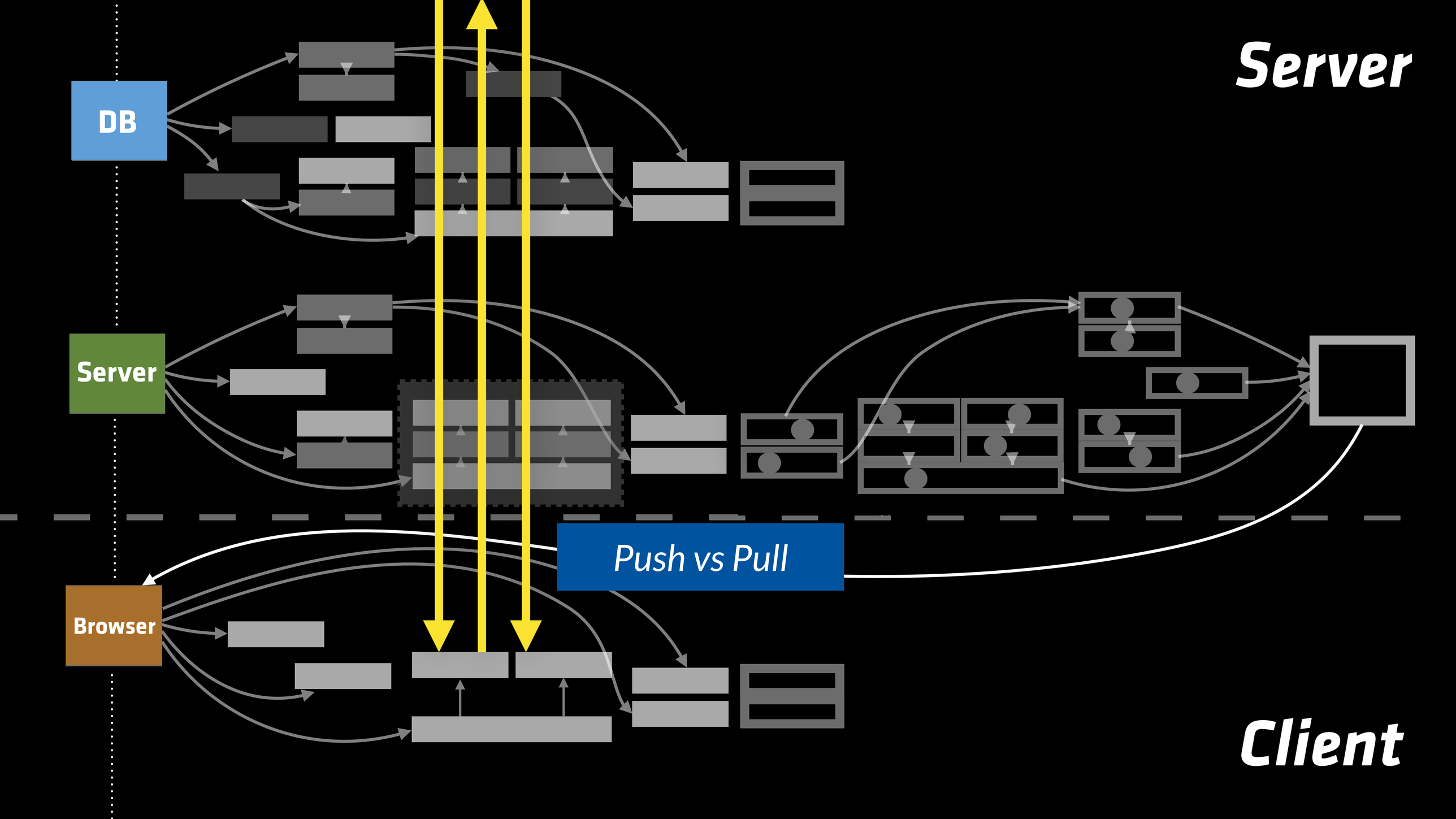

CPU
Same instance or different instance?
In incremental FP, this is NOT a no-op
CPU
[1, 2] + [3, 4, 5] + [] + [6, 7]
[1, 2, 3, 4, 5, 6, 7]
CPU
GPU
Requires
declarative signal()
GPU

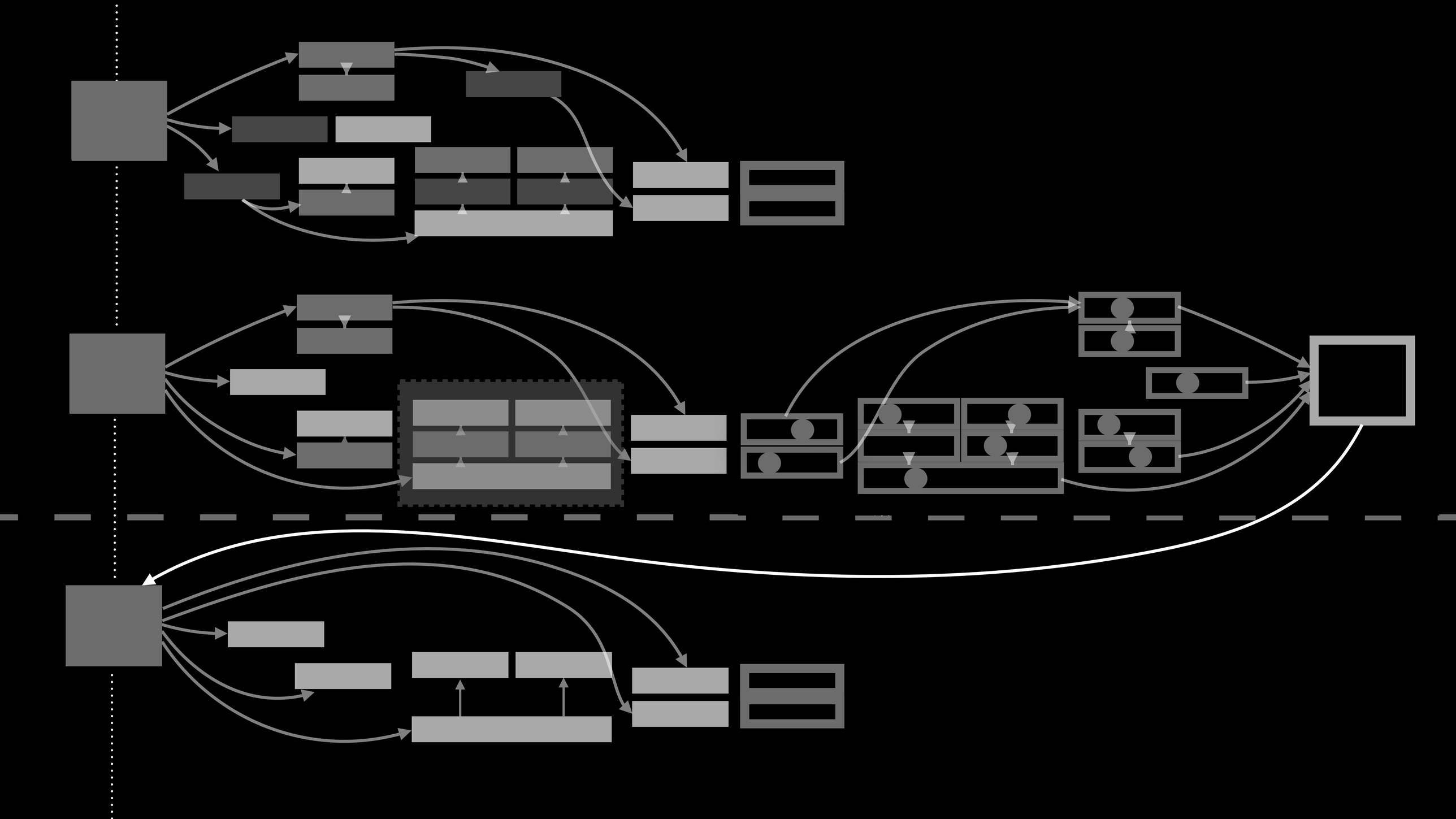

CPU
Same instance or different instance?
CPU
In incremental FP, this is NOT a no-op
[1, 2] + [3, 4, 5] + [] + [6, 7]
CPU
[1, 2, 3, 4, 5, 6, 7]
Determines whether output is same instance as before
GPU
Requires declarative signal()
GPU

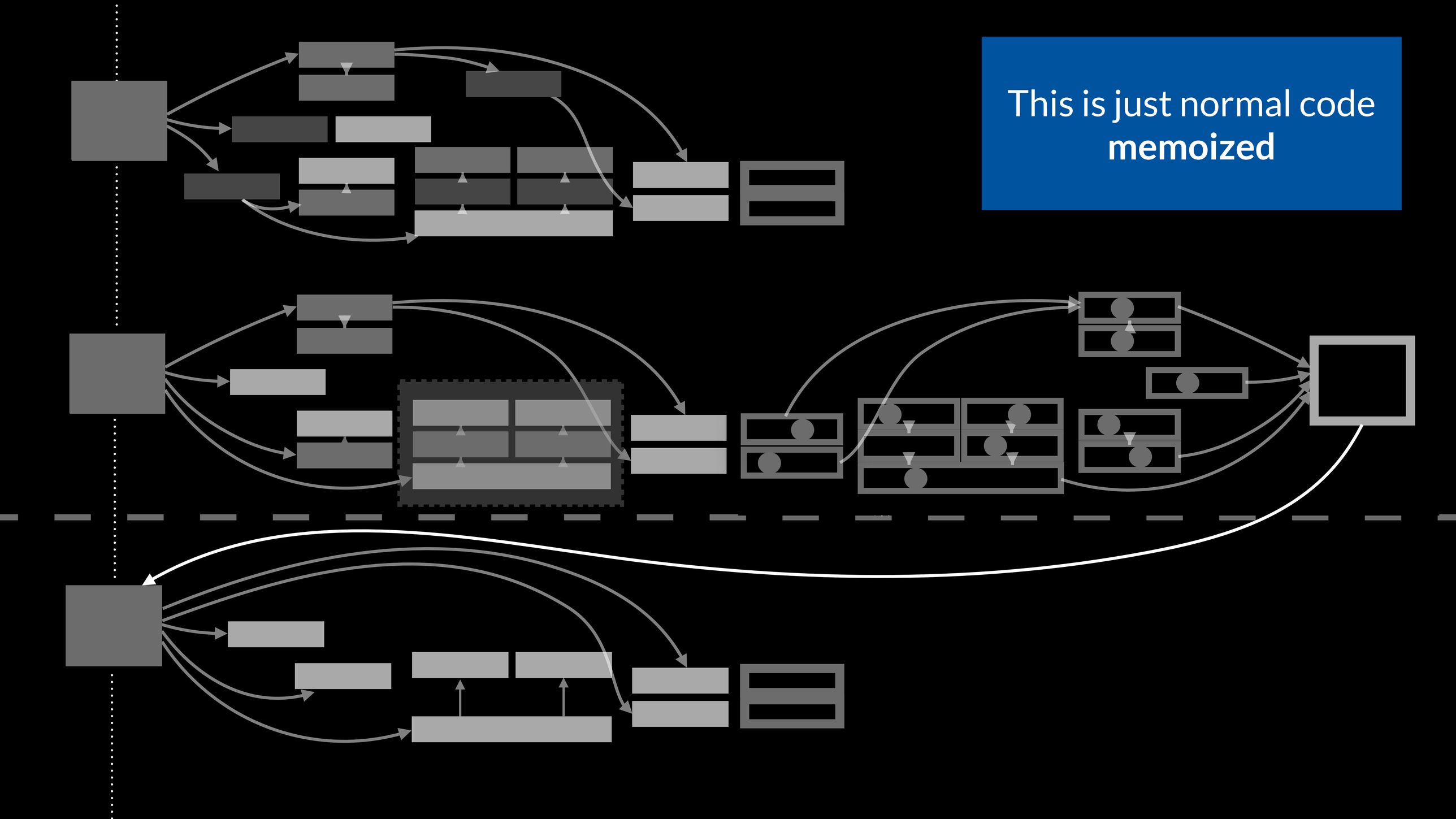

CPU
Same instance or different instance?
CPU
In incremental FP, this is NOT a no-op
[1, 2] + [3, 4, 5] + [] + [6, 7]
[1, 2, 3, 4, 5, 6, 7]
CPU
Determines whether output is same instance as before
Declarative signal()
= optimize away the no-no-op
GPU
Requires declarative signal()
GPU

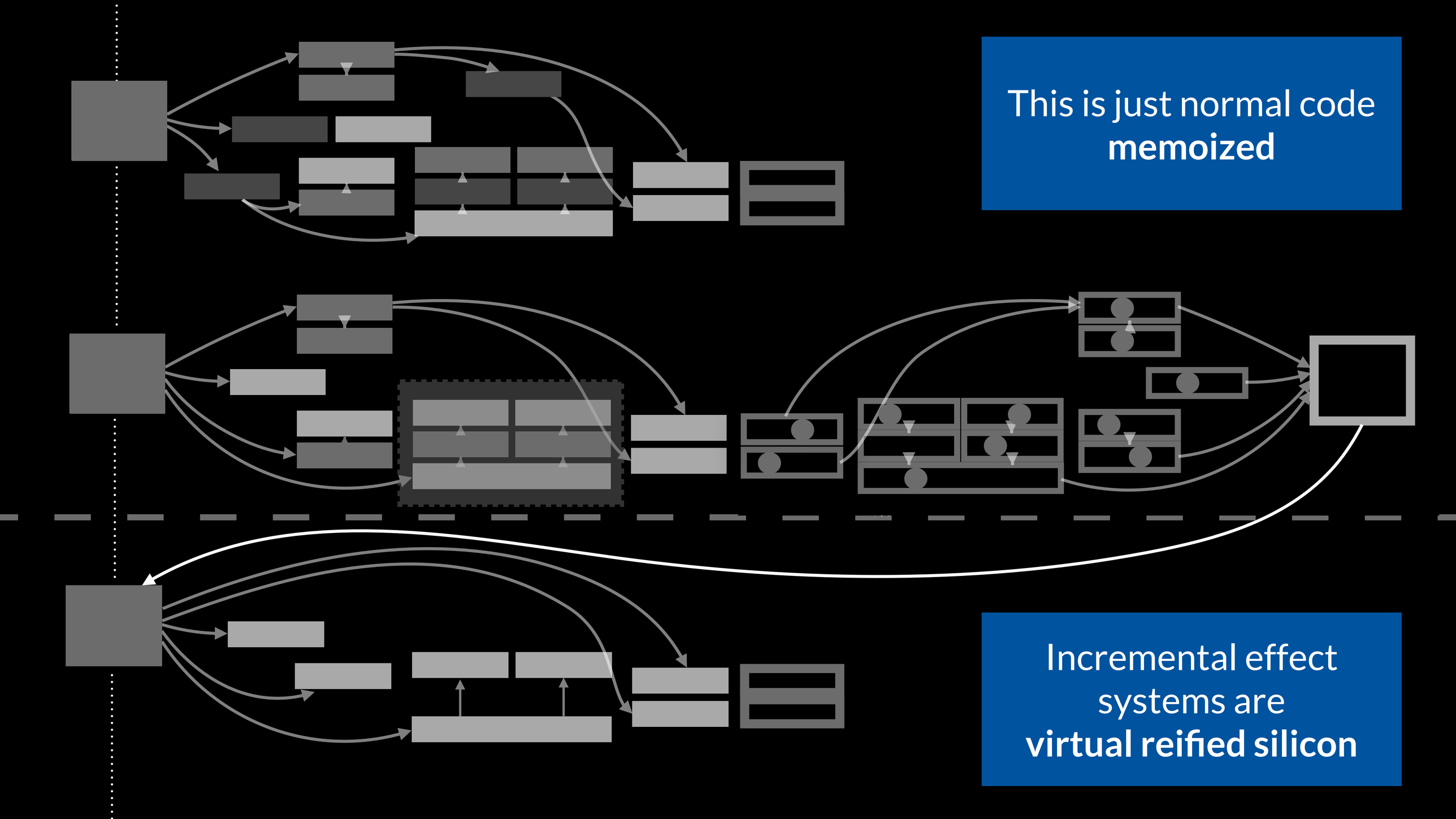

DB
Server
Browser

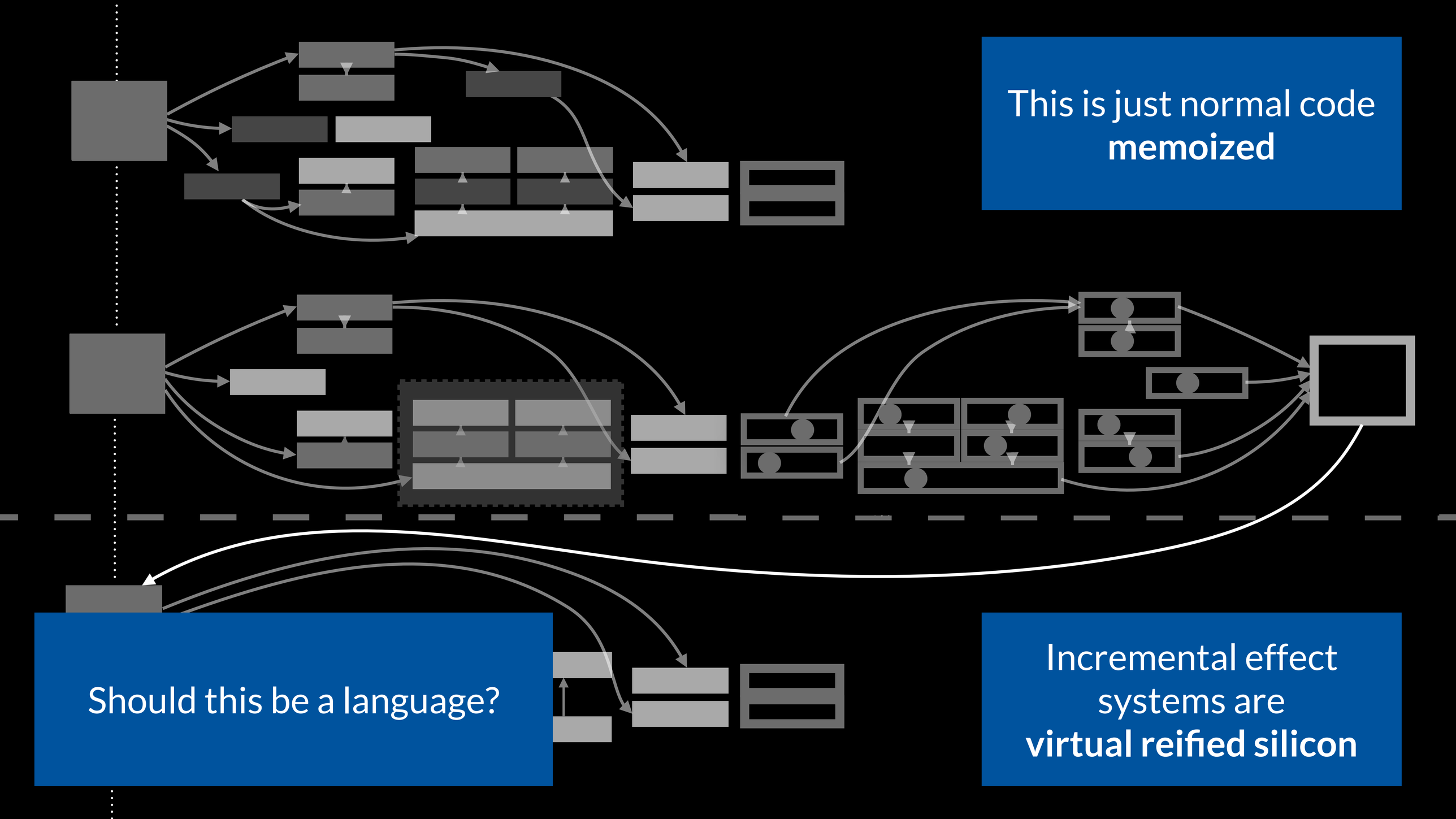

Server
DB
Server
Browser
Client


Server
DB
Server
Push vs Pull
Browser
Client

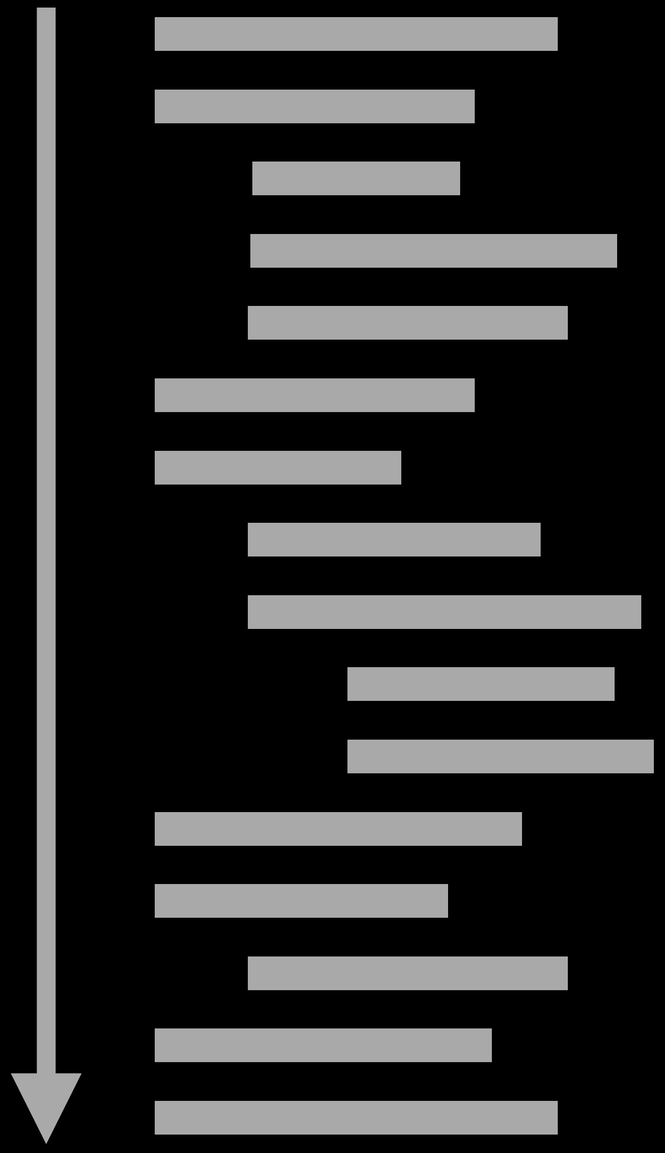



This is just normal code
memoized

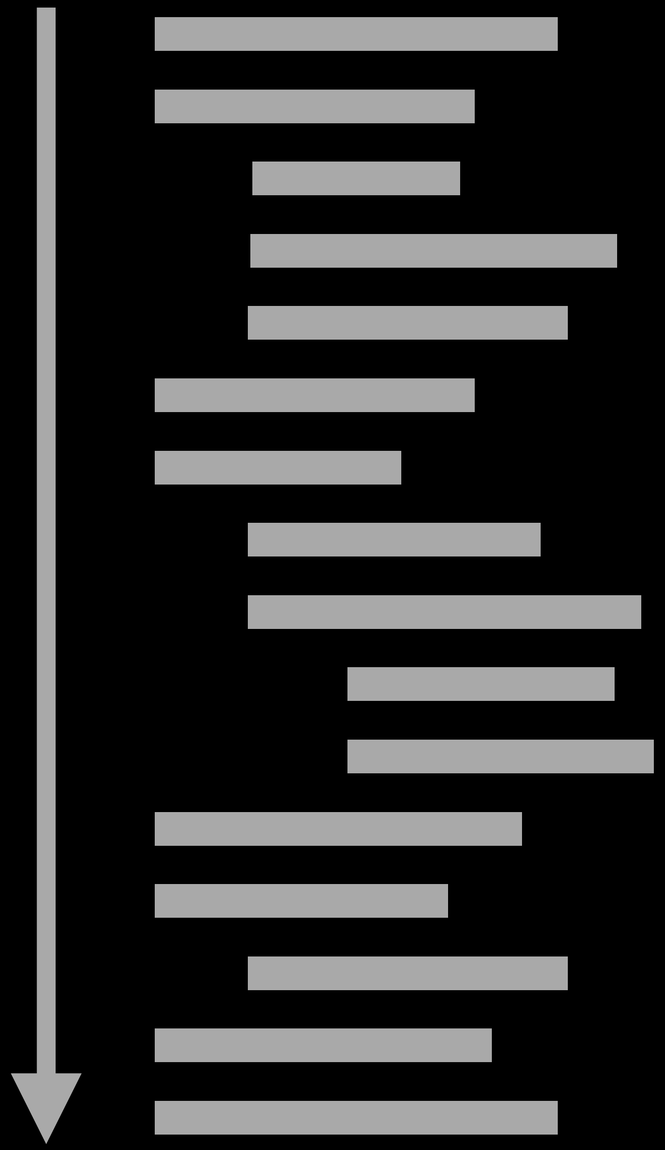

This is just normal code
memoized
Incremental effect
systems are
virtual reified silicon


This is just normal code
memoized
Should this be a language?
Incremental effect
systems are
virtual reified silicon

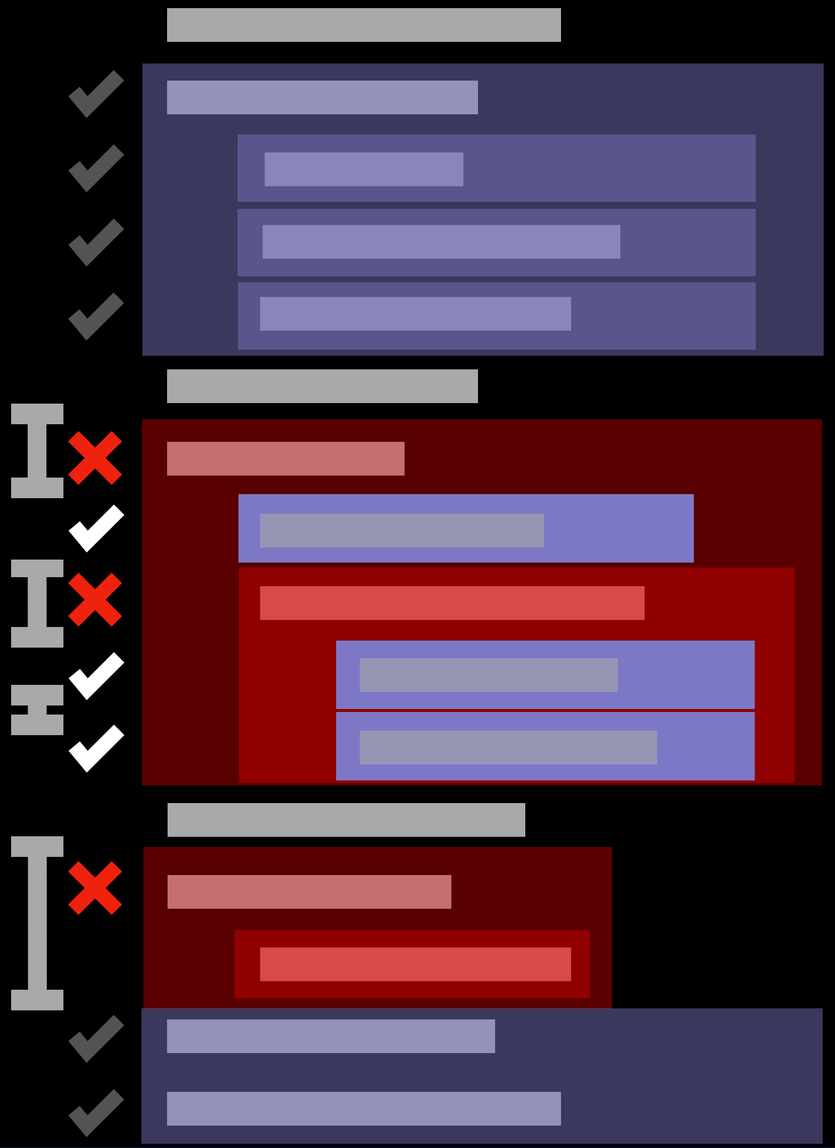
Use.GPU



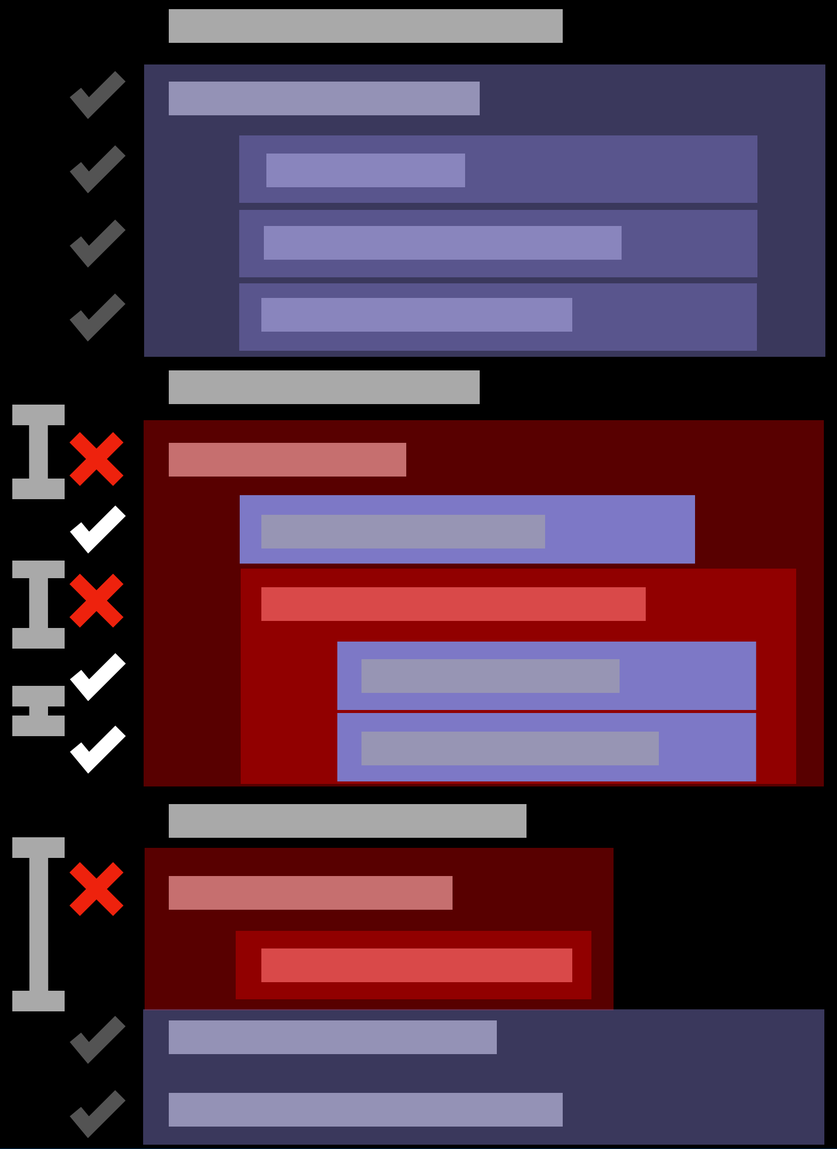
Write the typical code
to **render 1 frame in WebGPU**



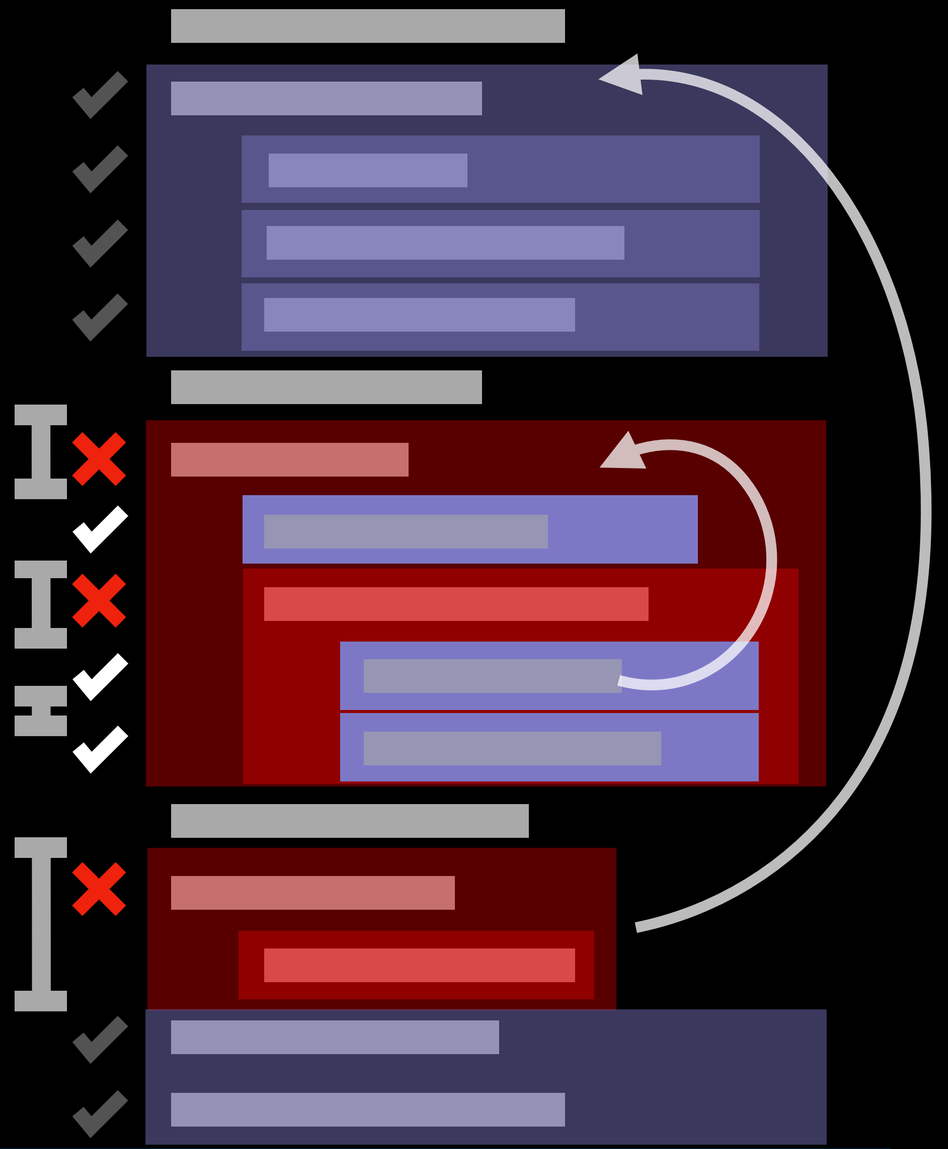
Write the typical code
to **render 1 frame in WebGPU**



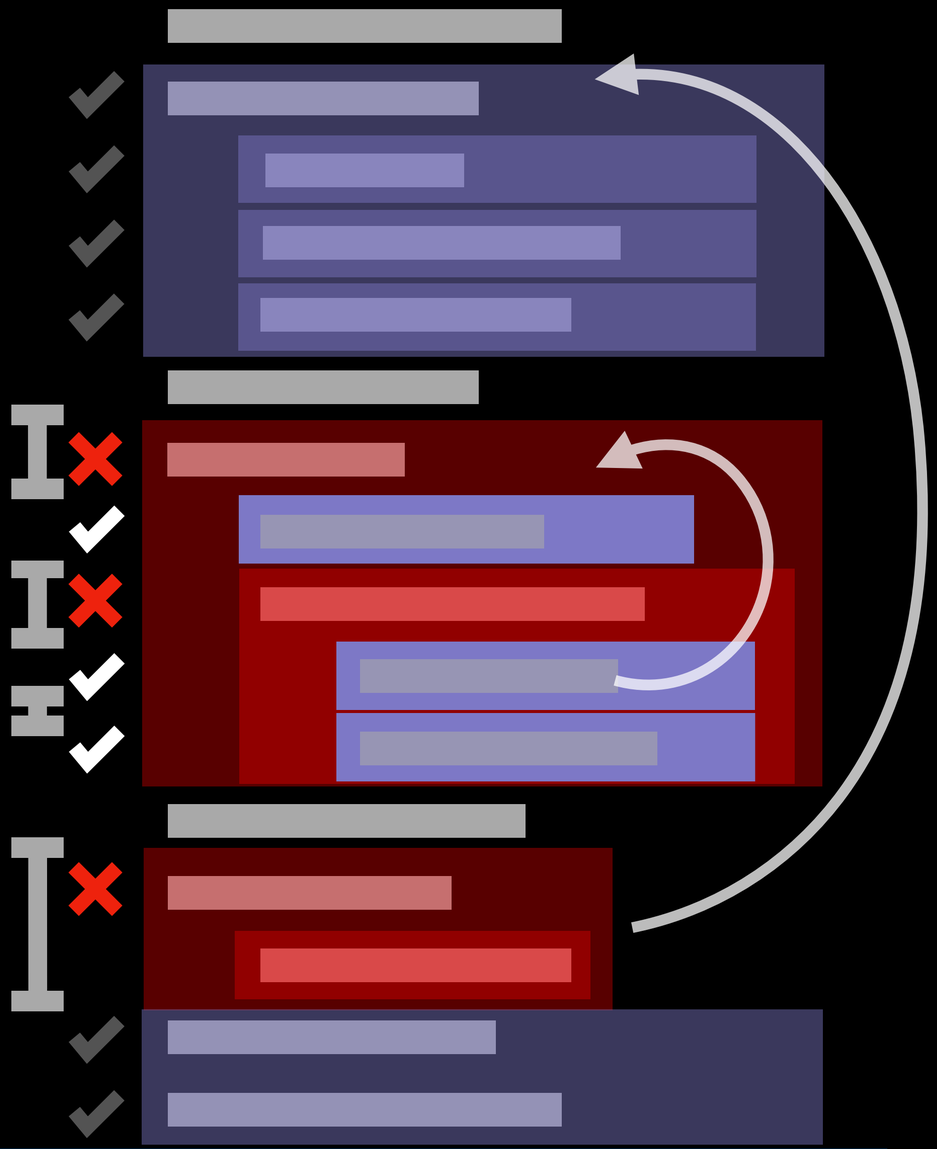
Memoize it using the
patterns of Live



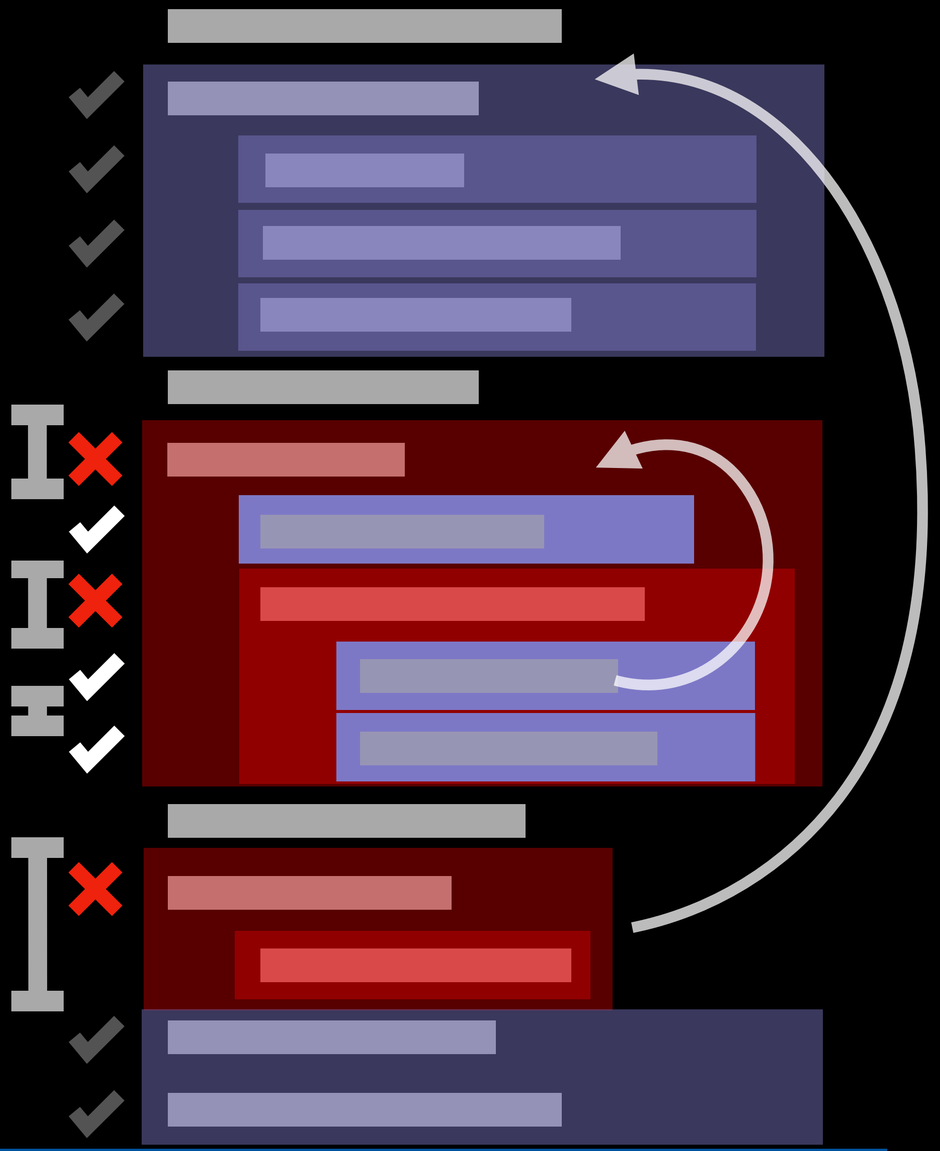
Memoize it using the
patterns of Live



Make it an **interactive app**
using **Live hooks**



Make it an **interactive app**
using **Live hooks**

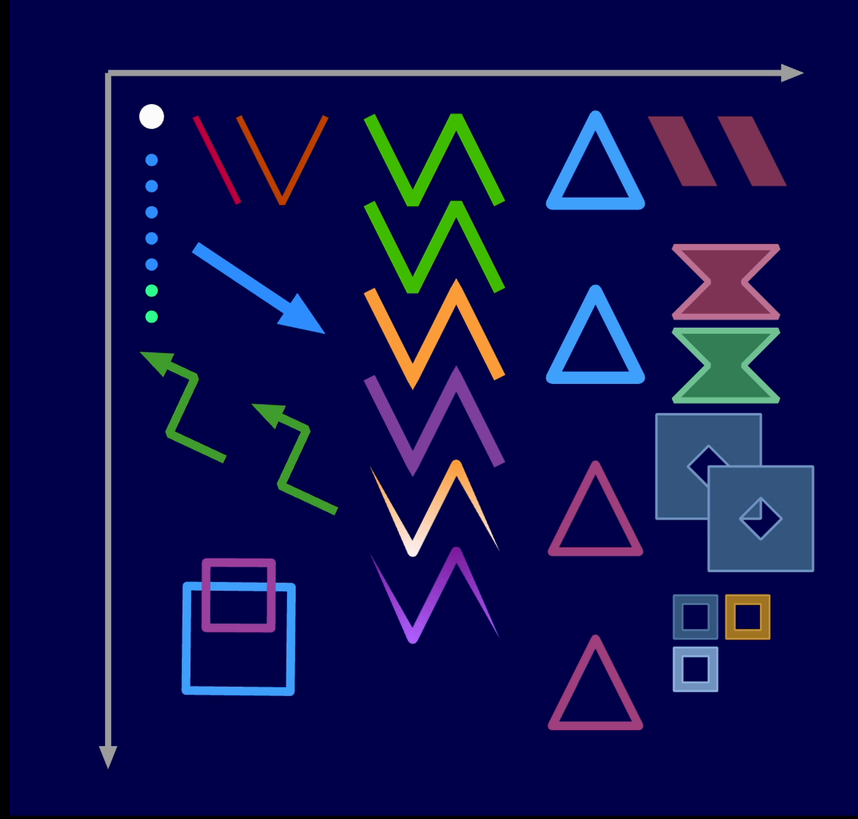


Make it an **interactive app** using **Live hooks**

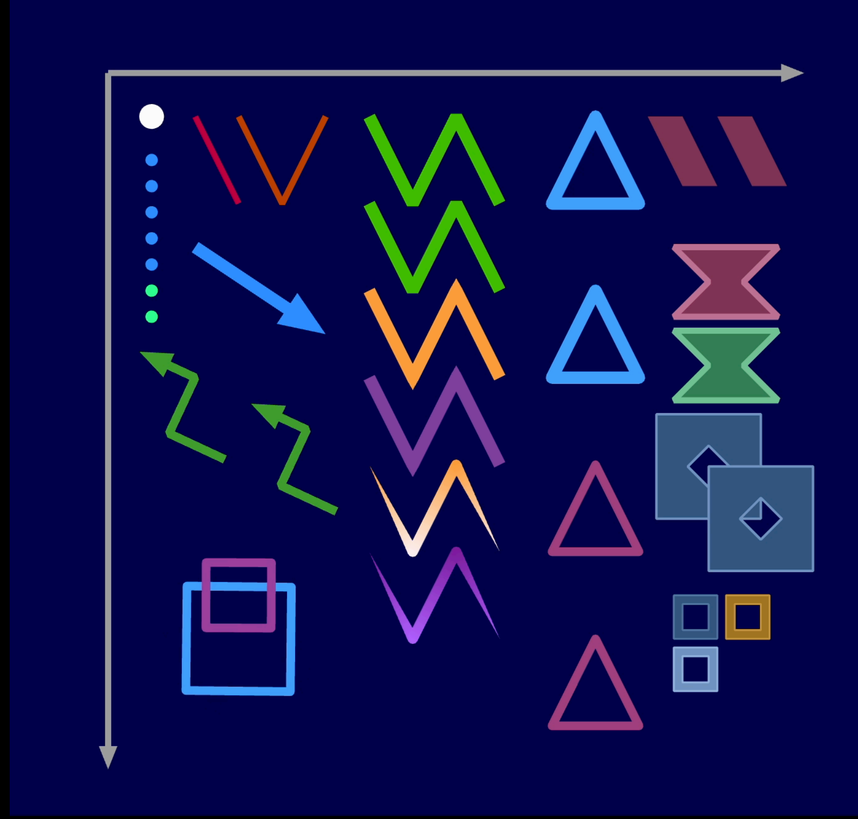
- OrbitCamera
- Provide(FrameContext)
- ViewProvider
  - Signal
  - Provide(ViewContext)
  - Provide(LayoutContext)
    - Memo(Cursor)
    - Memo(Pass)
    - Memo(ForwardRenderer)
    - Reconcile
      - Quote
        - Provide(VariantContext)
          - Data
            - Signal
            - Memo(RawLines)
          - Data
            - Signal
            - Memo(RawLines)
          - Data
            - Signal
            - Memo(ArrowLayer)
              - Memo(RawLines)
              - Memo(RawArrows)
      - Resolve(Quote)
    - Root(PassReconciler)
      - Provide(PassContext)
      - MultiGather
        - Unquote
          - SolidRender

Go **all the way**

"How do I draw a line?"

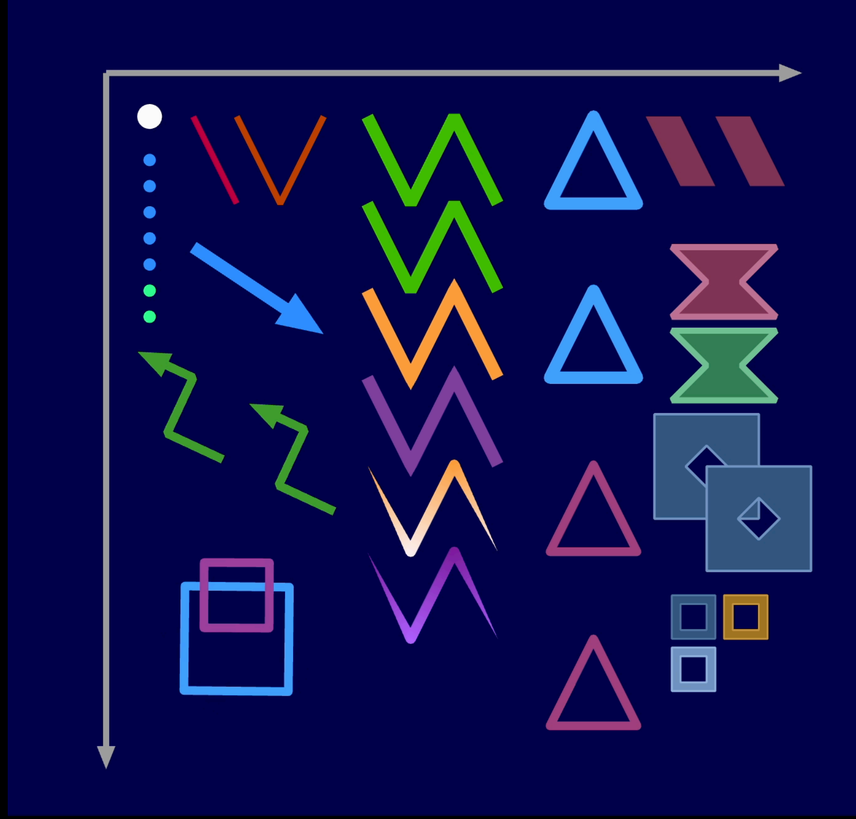


"How do I draw a line?"



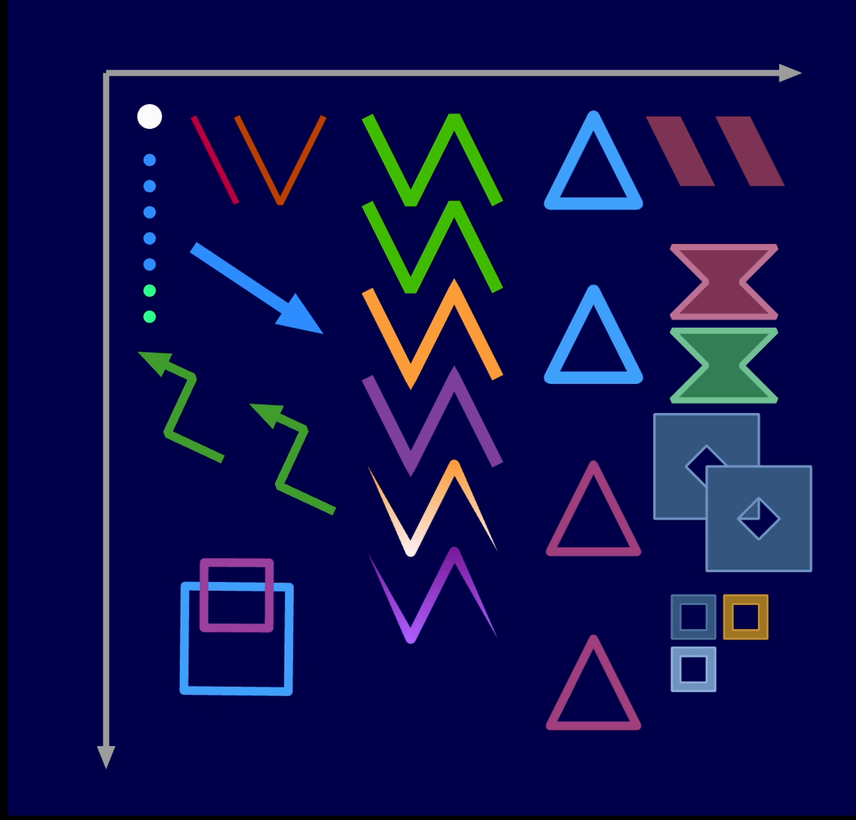
@use-gpu/plot

"How do I draw a line?"



@use-gpu/plot

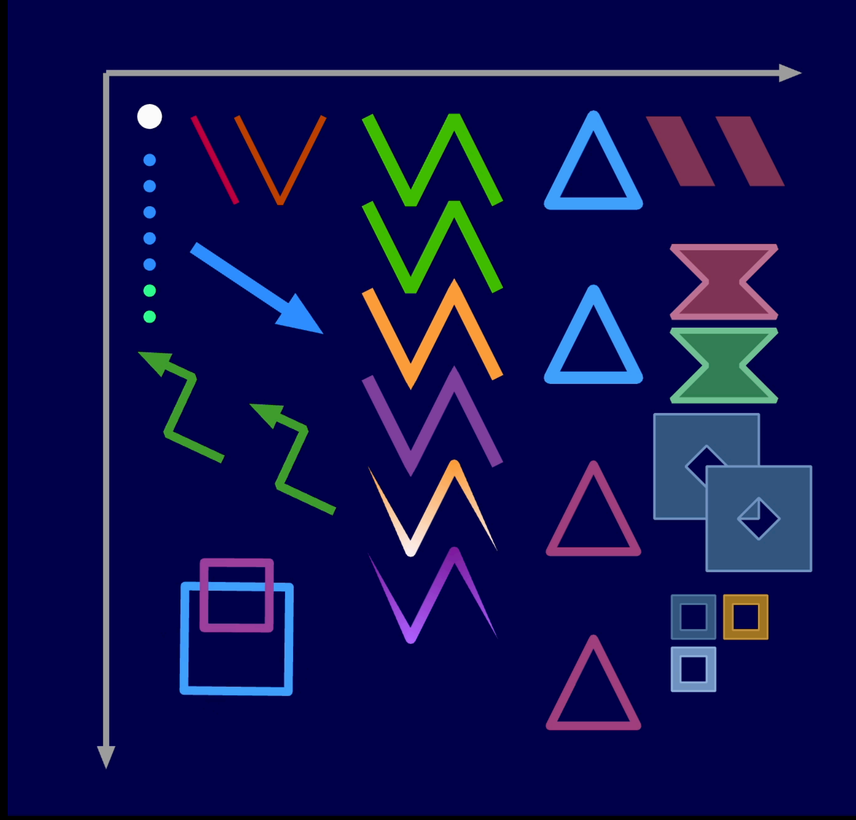
"How do I draw a line?"



```
<Line
  position={[[100, 50], [150, 150]]}
  width={5}
  color={'#c00040'}
/>

<Line
  position={[[150, 50], [200, 150], [250, 50]]}
  width={5}
  color={'#c04000'}
/>
```

@use-gpu/plot

"How do I draw a line?"

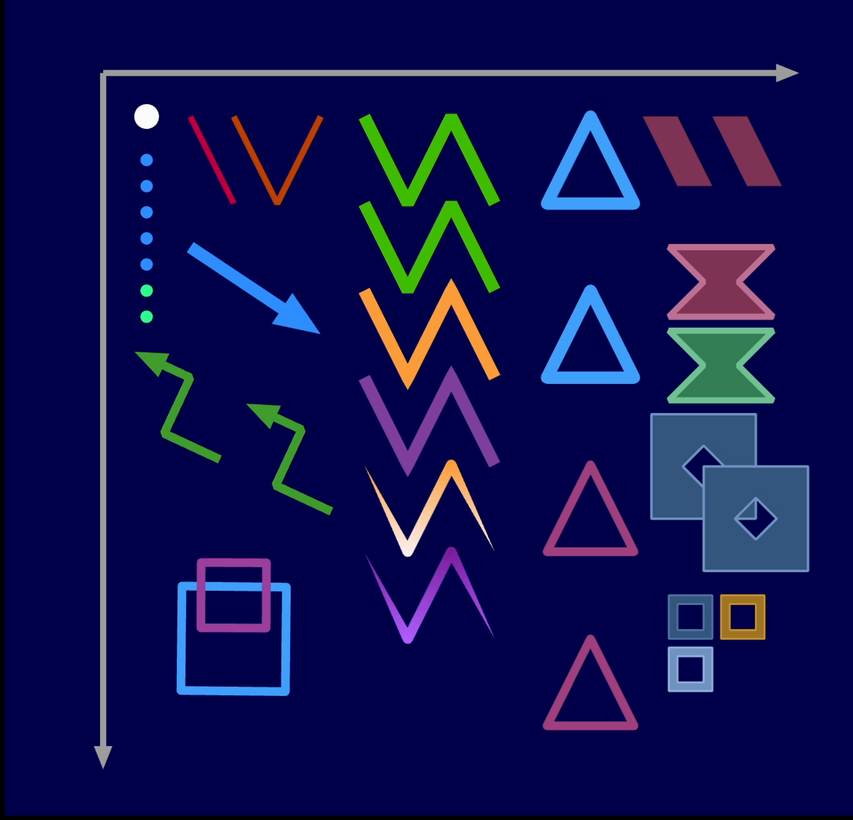


```
<Line
  position={[[100, 50], [150, 150]]}
  width={5}
  color={'#c00040'}
/>

<Line
  position={[[150, 50], [200, 150], [250, 50]]}
  width={5}
  color={'#c04000'}
/>
```

2D / 3D / 4D

@use-gpu/plot

"How do I draw many lines?"

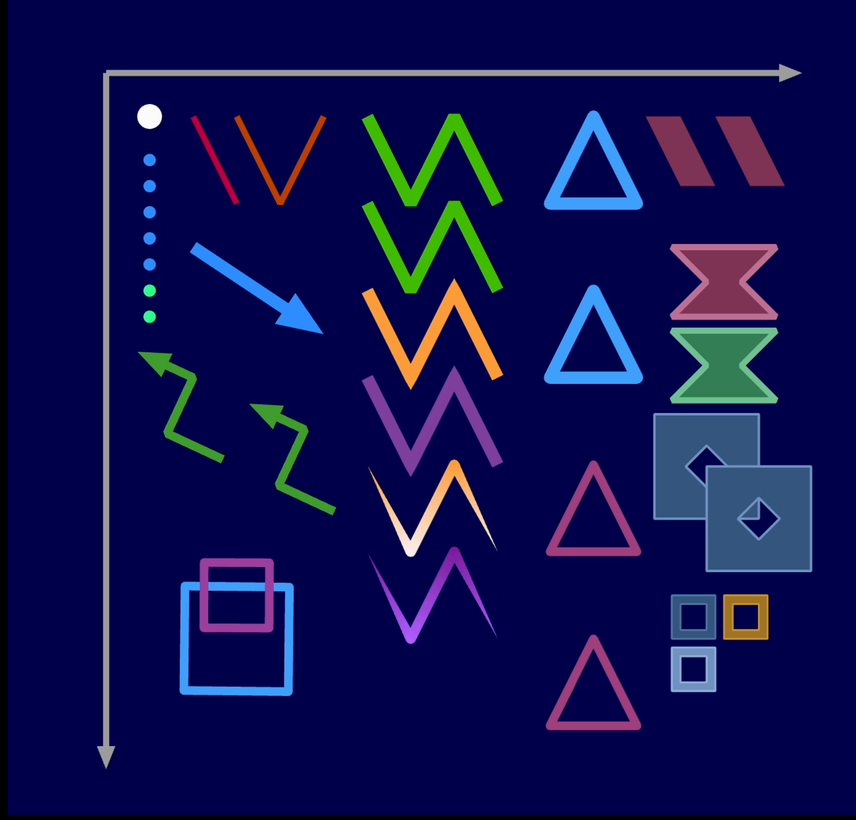


```
<Line
  positions={[
    [[300, 50], [350, 150], [400, 50], [450, 150]],
    [[300, 150], [350, 250], [400, 150], [450, 250]],
  ]}
  width={10}
  color={'#40c000'}
/>

<Line
  positions={[
    [[300, 250], [350, 350], [400, 250], [450, 350]],
    [[300, 350], [350, 450], [400, 350], [450, 450]],
  ]}
  width={10}
  color={['#ffa040', '#7f40a0']}
  join="miter"
/>
```

@use-gpu/plot

"How do I draw many lines?"

```
<Line
  positions={[
    [[300, 50], [350, 150], [400, 50], [450, 150]],
    [[300, 150], [350, 250], [400, 150], [450, 250]],
  ]}
  width={10}
  color={'#40c000'}
/>

<Line
  positions={[
    [[300, 250], [350, 350], [400, 250], [450, 350]],
    [[300, 350], [350, 450], [400, 350], [450, 450]],
  ]}
  width={10}
  color={['#ffa040', '#7f40a0']}
  join="miter"
/>
```

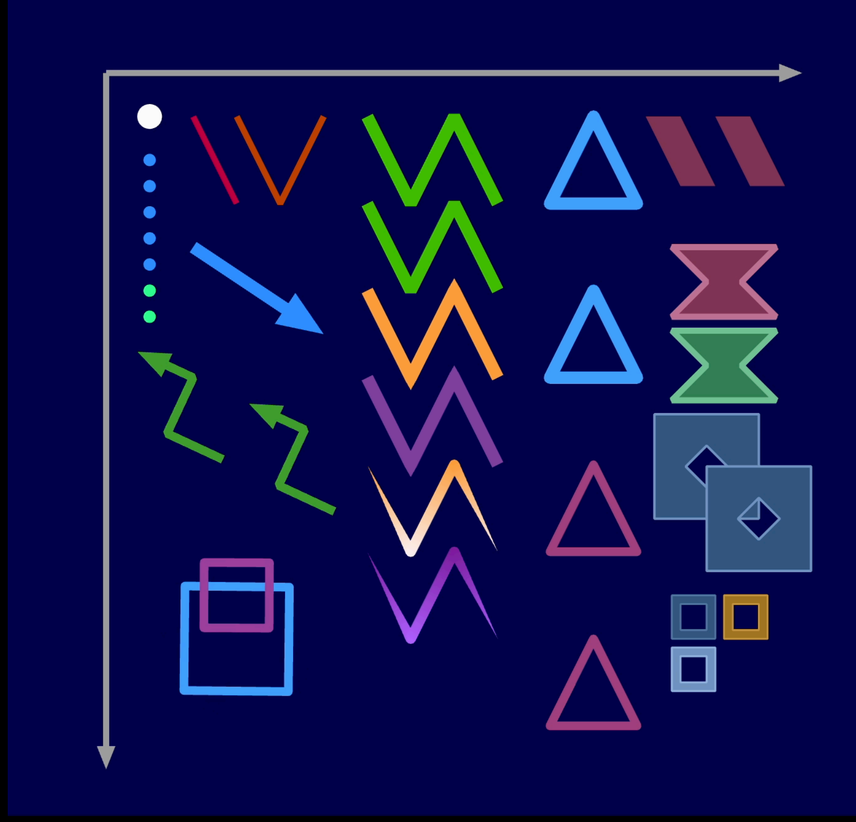


Design **sane APIs**
for the common use cases

@use-gpu/plot

"How do I draw many lines?"

```
<Line
  positions={[
    [[300, 50], [350, 150], [400, 50], [450, 150]],
    [[300, 150], [350, 250], [400, 150], [450, 250]],
  ]}
  width={10}
  color={'#40c000'}
/>

<Line
  positions={[
    [[300, 250], [350, 350], [400, 250], [450, 350]],
    [[300, 350], [350, 450], [400, 350], [450, 450]],
  ]}
  width={10}
  color={['#ffa040', '#7f40a0']}
  join="miter"
/>
```



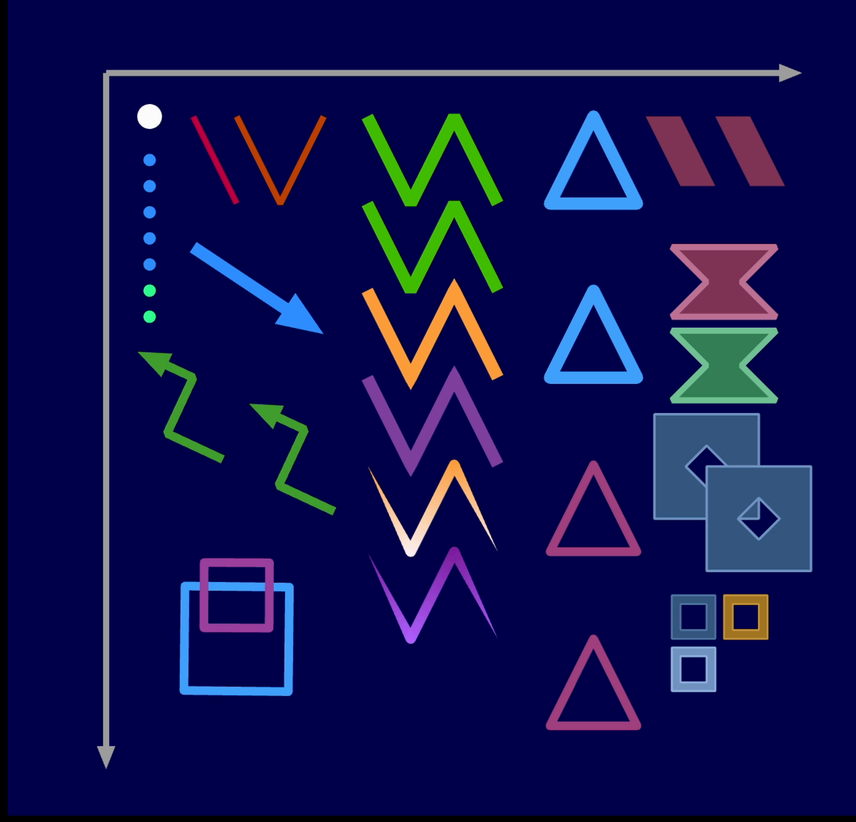
Design **sane APIs**
for the common use cases

Provide zero-cost
escape hatches for the rest

@use-gpu/plot

"How do I draw many lines?"

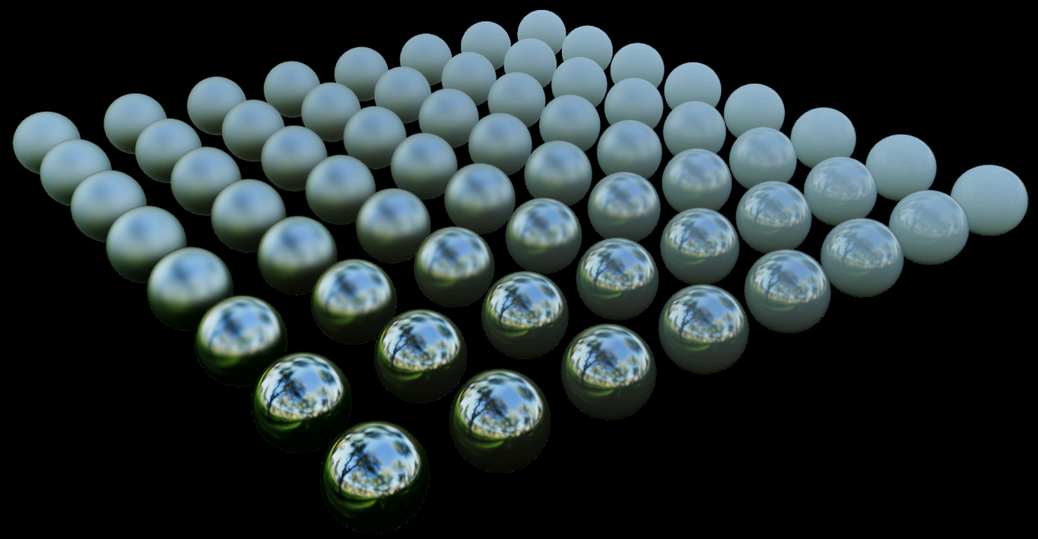
```
<Line
  positions={[
    [[300, 50], [350, 150], [400, 50], [450, 150]],
    [[300, 150], [350, 250], [400, 150], [450, 250]],
  ]}
  width={10}
  color={'#40c000'}
/>

<Line
  positions={[
    [[300, 250], [350, 350], [400, 250], [450, 350]],
    [[300, 350], [350, 450], [400, 350], [450, 450]],
  ]}
  width={10}
  color={['#ffa040', '#7f40a0']}
  join="miter"
/>
```



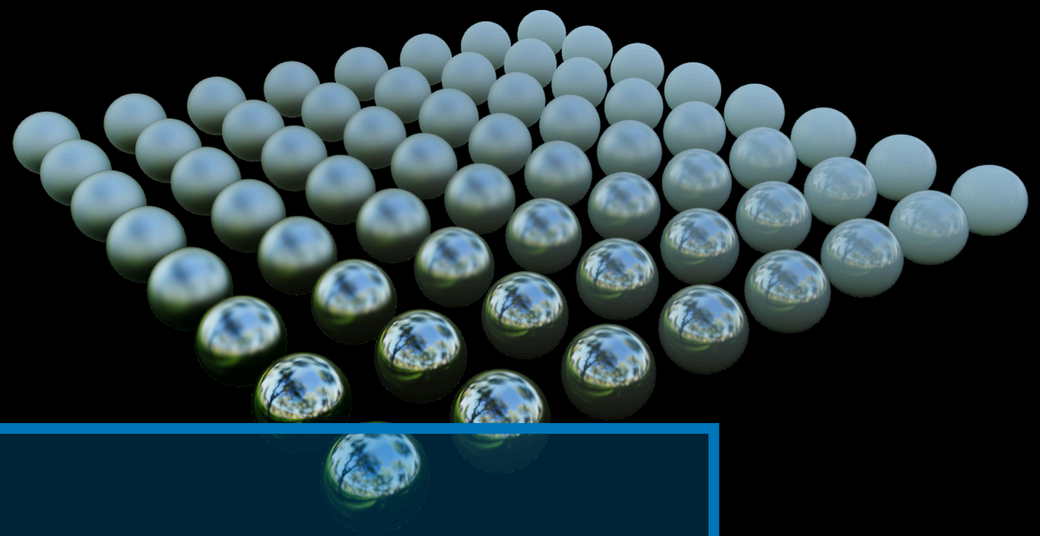
Design **sane APIs**
for the common use cases

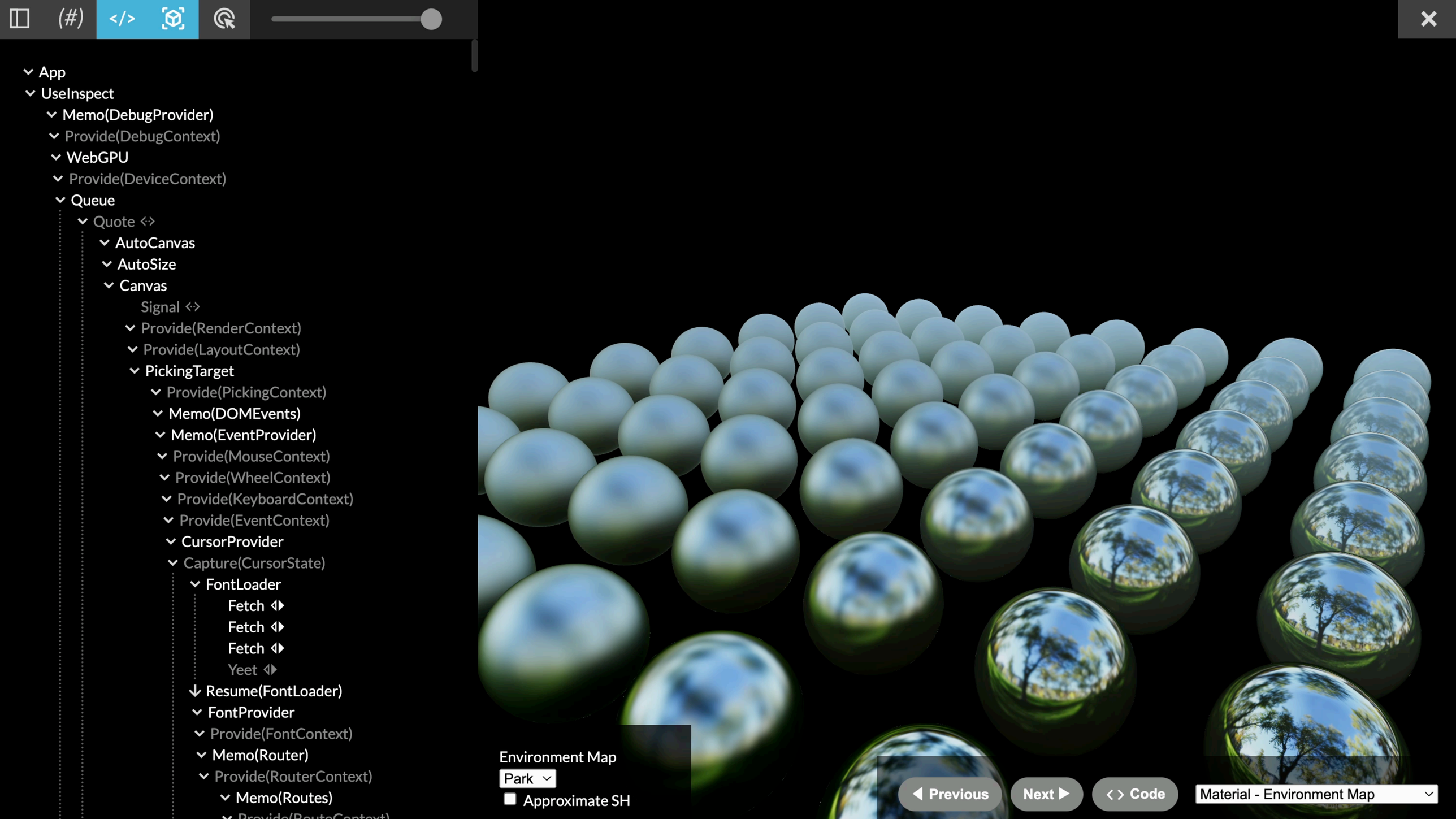
Provide zero-cost
escape hatches for the rest

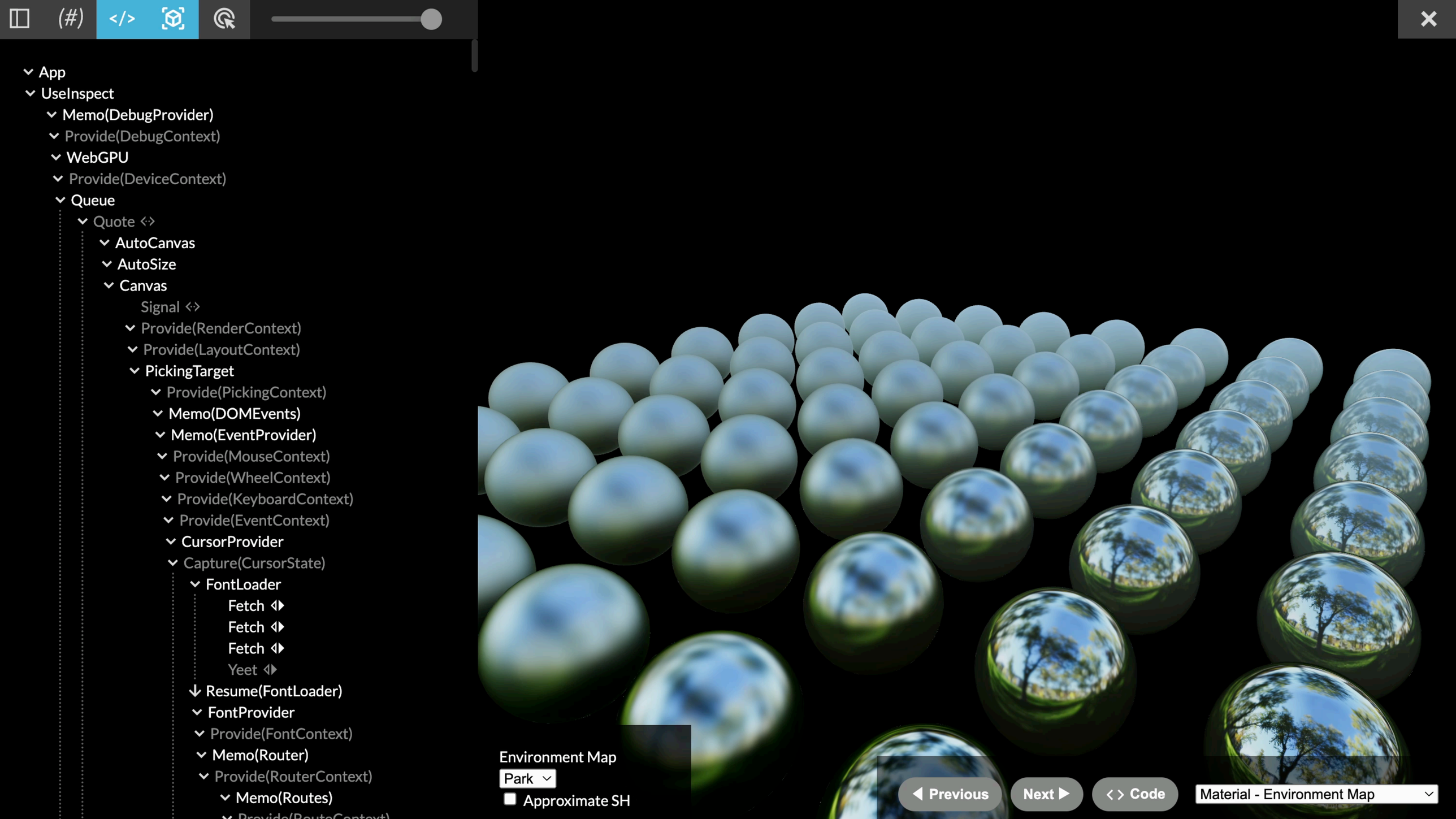
```
<Camera active={!zooming && !panning}>
  <PrefilteredEnvMap
    texture={texture}
    gain={1}
    seamFix={seamFix}
    debugGrid={debugGrid}
  >{
    (cubeMap, texture) =>
      <LinearRGB tonemap="aces" gain={3}>
        <Cursor cursor='move' />
        <Pass lights>

          <Environment map={cubeMap} preset={envPreset}>
            <Scene>
              {
                seq(8).flatMap(i =>
                  seq(8).map(j => (
                    <Node key={`${i}-${j}`} position={[i - 3.5, 0, j - 3.5]} scale={[0.35, 0.35, 0.35]}>
                      <PBRMaterial
                        albedo={[0.6, 0.85, 1.0, 1.0]}
                        metalness={i / 7}
                        roughness={j / 7}
                      >
                        <Mesh mesh={mesh} shaded />
                      </PBRMaterial>
                    </Node>
                  ))
                )
              }
            </Scene>
          </Environment>

        </Pass>
```



```
<Camera active={!zooming && !panning}>
  <PrefilteredEnvMap
    texture={texture}
    gain={1}
    seamFix={seamFix}
    debugGrid={debugGrid}
  >{
    (cubeMap, texture) =>
      <LinearRGB tonemap="aces" gain={3}>
        <Cursor cursor='move' />
        <Pass lights>
          <Environment map={cubeMap} preset={envPreset}>
            <Scene>
              {
                seq(8).flatMap(i =>
                  seq(8).map(j => (
                    <Node key={`${i}-${j}`} position={[i - 3.5, 0, j - 3.5]} scale={[0.35, 0.35, 0.35]}>
                      <PBRMaterial
                        albedo={[0.6, 0.85, 1.0, 1.0]}
                        metalness={i / 7}
                        roughness={j / 7}
                      >
                        <Mesh mesh={mesh} shaded />
                      </PBRMaterial>
                    </Node>
                  ))
                )
              }
            </Scene>
          </Environment>

        </Pass>
```

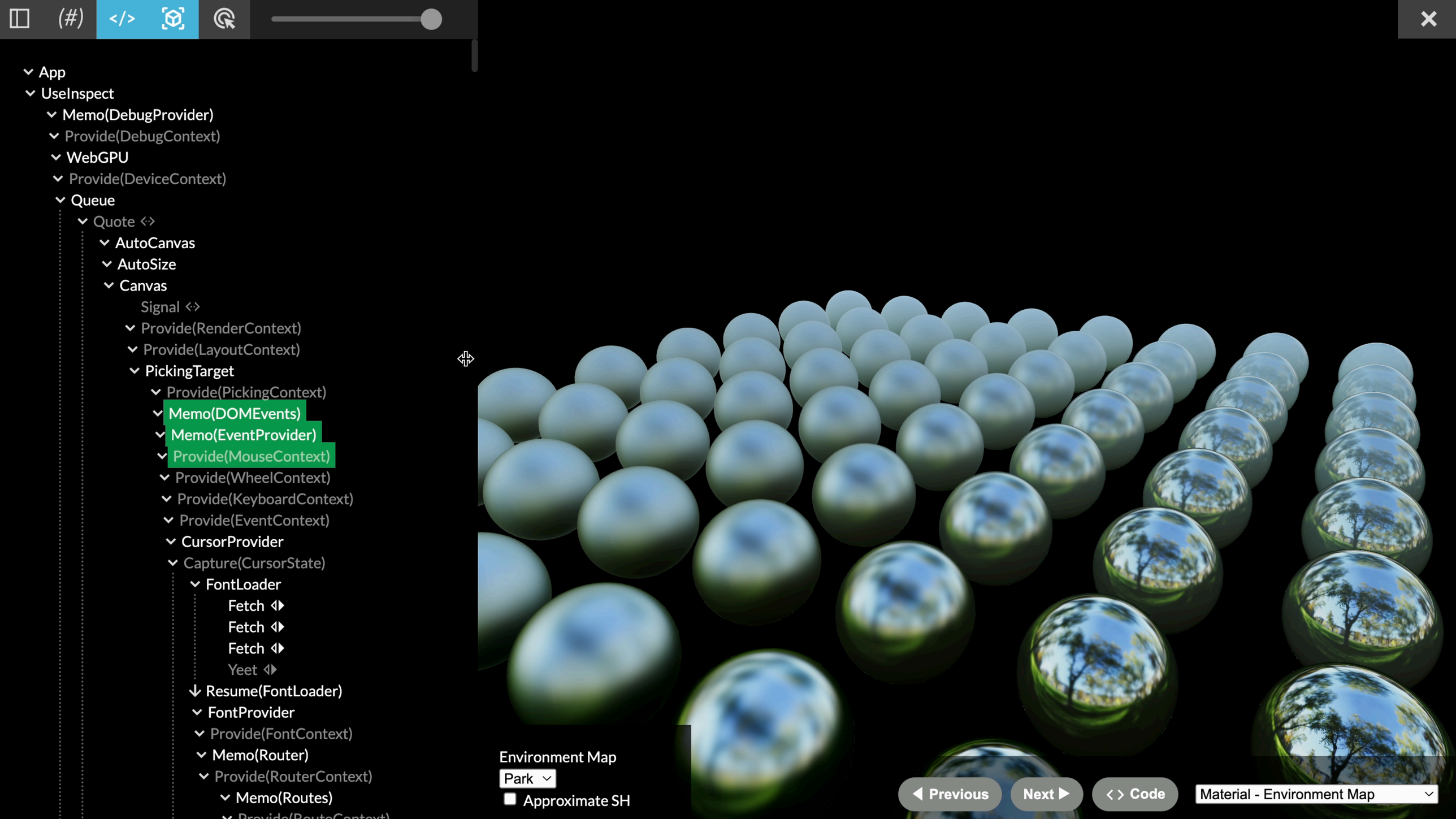


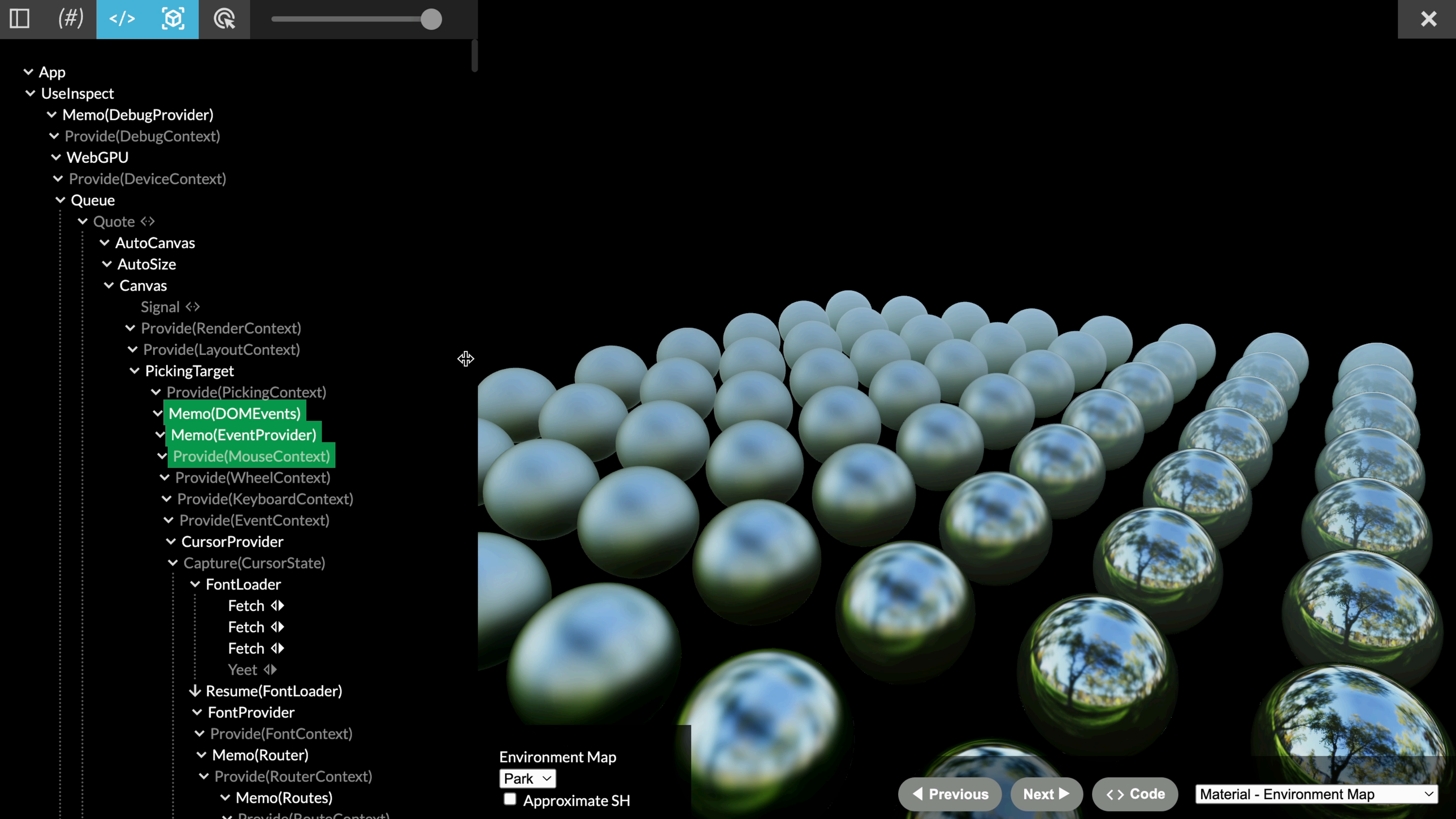

App
UseInspect
Memo(DebugProvider)
Provide(DebugContext)
WebGPU
Provide(DeviceContext)
Queue
Quote
AutoCanvas
AutoSize
Canvas
Signal
Provide(RenderContext)
Provide(LayoutContext)
PickingTarget
Provide(PickingContext)
Memo(DOMEvents)
Memo(EventProvider)
Provide(MouseContext)
Provide(WheelContext)
Provide(KeyboardContext)
Provide(EventContext)
CursorProvider
Capture(CursorState)
FontLoader
Fetch
Fetch
Fetch
Yeet
Resume(FontLoader)
FontProvider
Provide(FontContext)
Memo(Router)
Provide(RouterContext)
Memo(Routes)
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

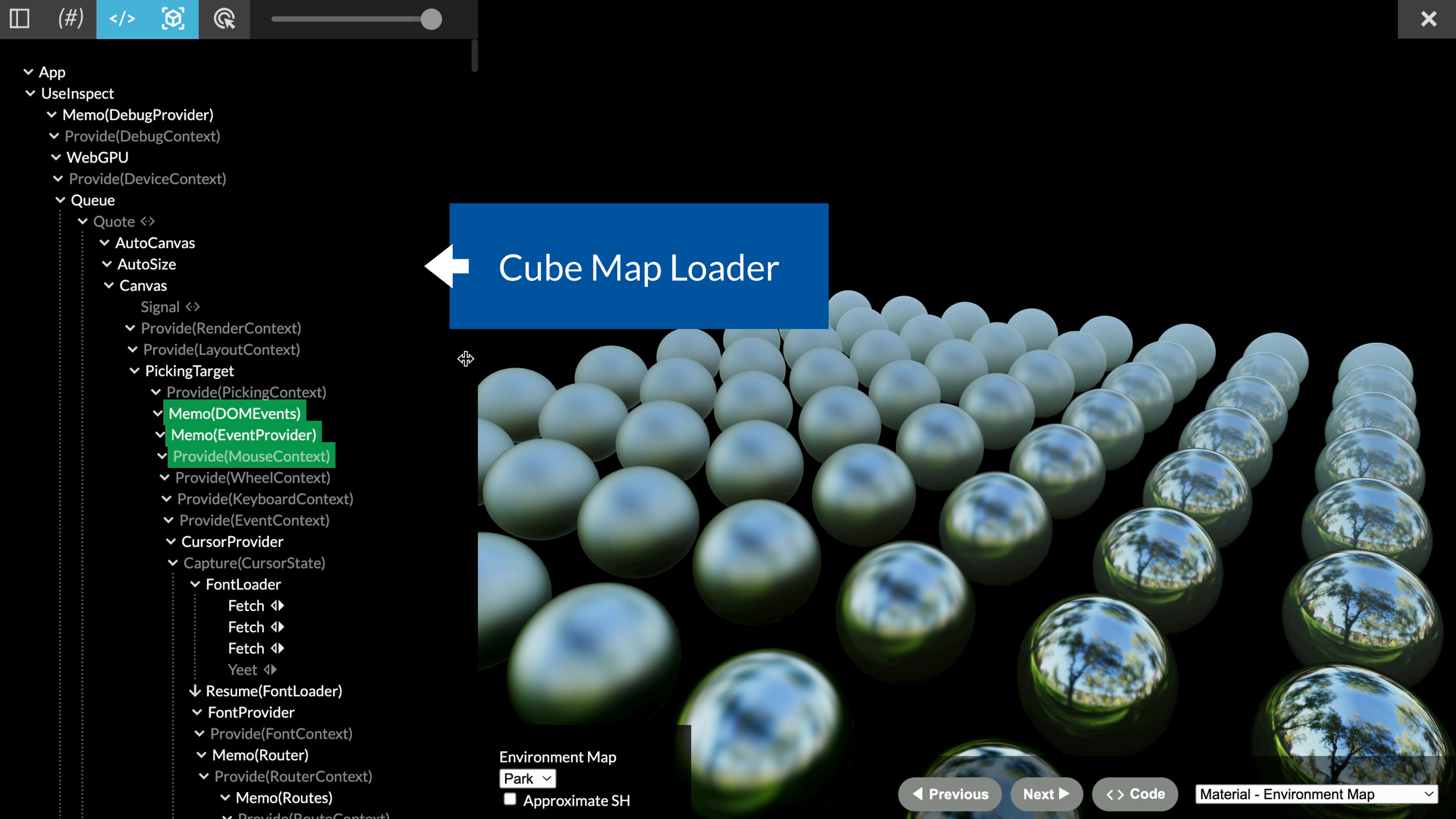

App
UseInspect
Memo(DebugProvider)
Provide(DebugContext)
WebGPU
Provide(DeviceContext)
Queue
Quote
AutoCanvas
AutoSize
Canvas
Signal
Provide(RenderContext)
Provide(LayoutContext)
PickingTarget
Provide(PickingContext)
Memo(DOMEvents)
Memo(EventProvider)
Provide(MouseContext)
Provide(WheelContext)
Provide(KeyboardContext)
Provide(EventContext)
CursorProvider
Capture(CursorState)
FontLoader
Fetch
Fetch
Fetch
Yeet
Resume(FontLoader)
FontProvider
Provide(FontContext)
Memo(Router)
Provide(RouterContext)
Memo(Routes)
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

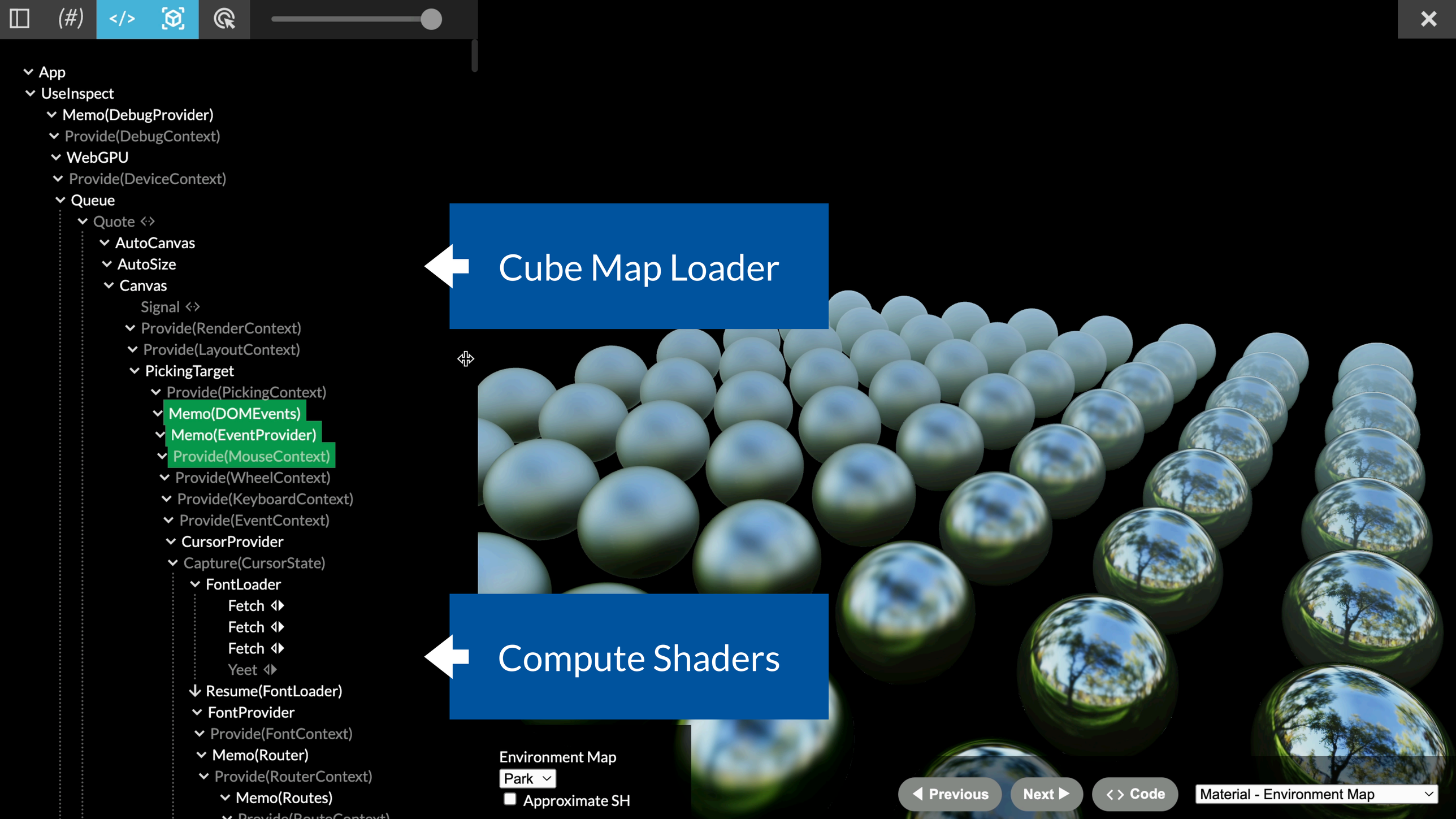

App
UseInspect
Memo(DebugProvider)
Provide(DebugContext)
WebGPU
Provide(DeviceContext)
Queue
Quote
AutoCanvas
AutoSize
Canvas
Signal
Provide(RenderContext)
Provide(LayoutContext)
PickingTarget
Provide(PickingContext)
Memo(DOMEvents)
Memo(EventProvider)
Provide(MouseContext)
Provide(WheelContext)
Provide(KeyboardContext)
Provide(EventContext)
CursorProvider
Capture(CursorState)
FontLoader
Fetch
Fetch
Fetch
Yeet
Resume(FontLoader)
FontProvider
Provide(FontContext)
Memo(Router)
Provide(RouterContext)
Memo(Routes)
WebGPU Canvas
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

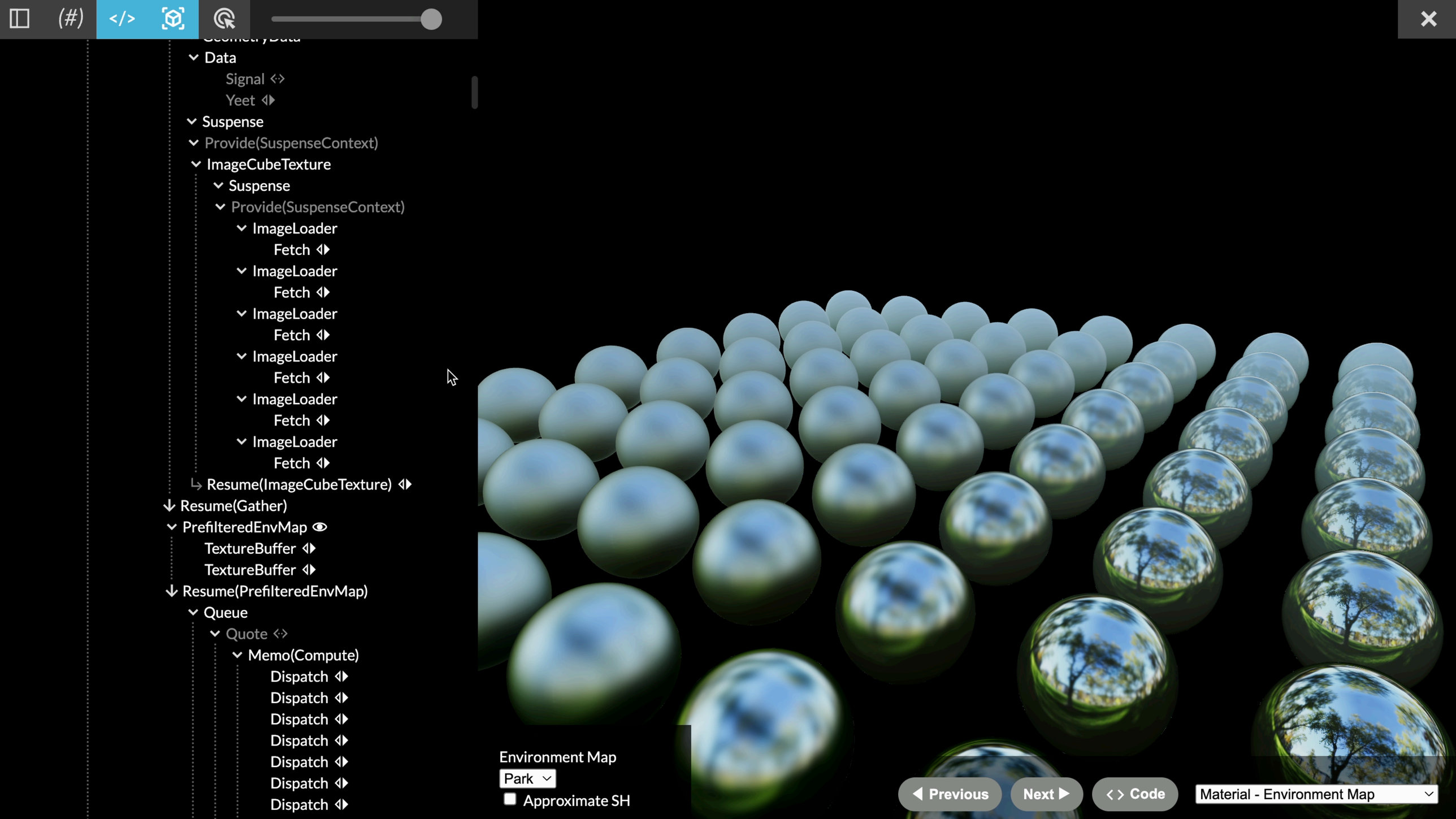

App
UseInspect
Memo(DebugProvider)
Provide(DebugContext)
WebGPU
Provide(DeviceContext)
Queue
Quote
AutoCanvas
AutoSize
Canvas
Signal
Provide(RenderContext)
Provide(LayoutContext)
PickingTarget
Provide(PickingContext)
Memo(DOMEvents)
Memo(EventProvider)
Provide(MouseContext)
Provide(WheelContext)
Provide(KeyboardContext)
Provide(EventContext)
CursorProvider
Capture(CursorState)
FontLoader
Fetch
Fetch
Fetch
Yeet
Resume(FontLoader)
FontProvider
Provide(FontContext)
Memo(Router)
Provide(RouterContext)
Memo(Routes)
WebGPU Canvas
Event handler
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

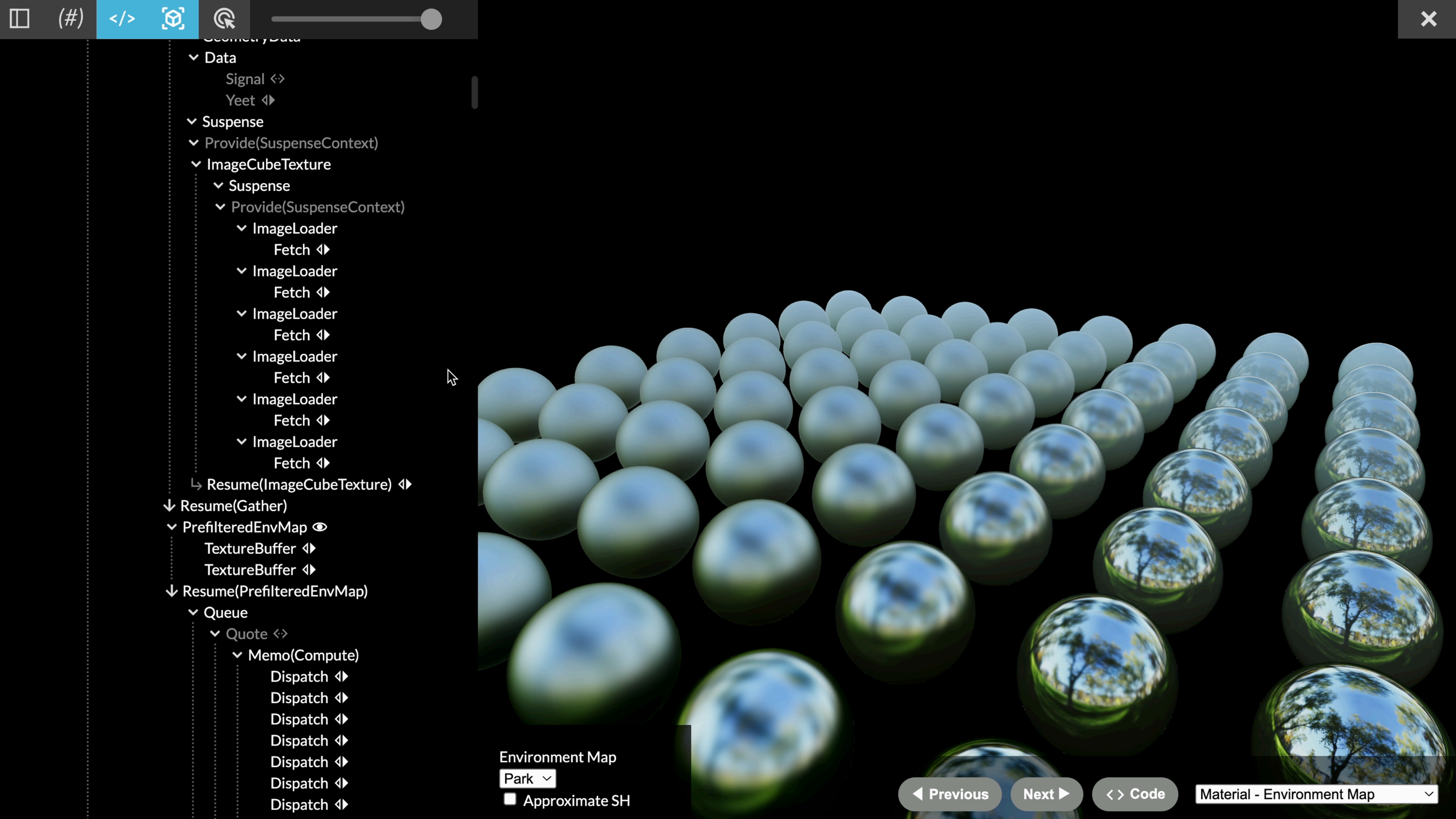

App
UseInspect
Memo(DebugProvider)
Provide(DebugContext)
WebGPU
Provide(DeviceContext)
Queue
Quote
AutoCanvas
AutoSize
Canvas
Signal
Provide(RenderContext)
Provide(LayoutContext)
PickingTarget
Provide(PickingContext)
Memo(DOMEvents)
Memo(EventProvider)
Provide(MouseContext)
Provide(WheelContext)
Provide(KeyboardContext)
Provide(EventContext)
CursorProvider
Capture(CursorState)
FontLoader
Fetch
Fetch
Fetch
Yeet
Resume(FontLoader)
FontProvider
Provide(FontContext)
Memo(Router)
Provide(RouterContext)
Memo(Routes)
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

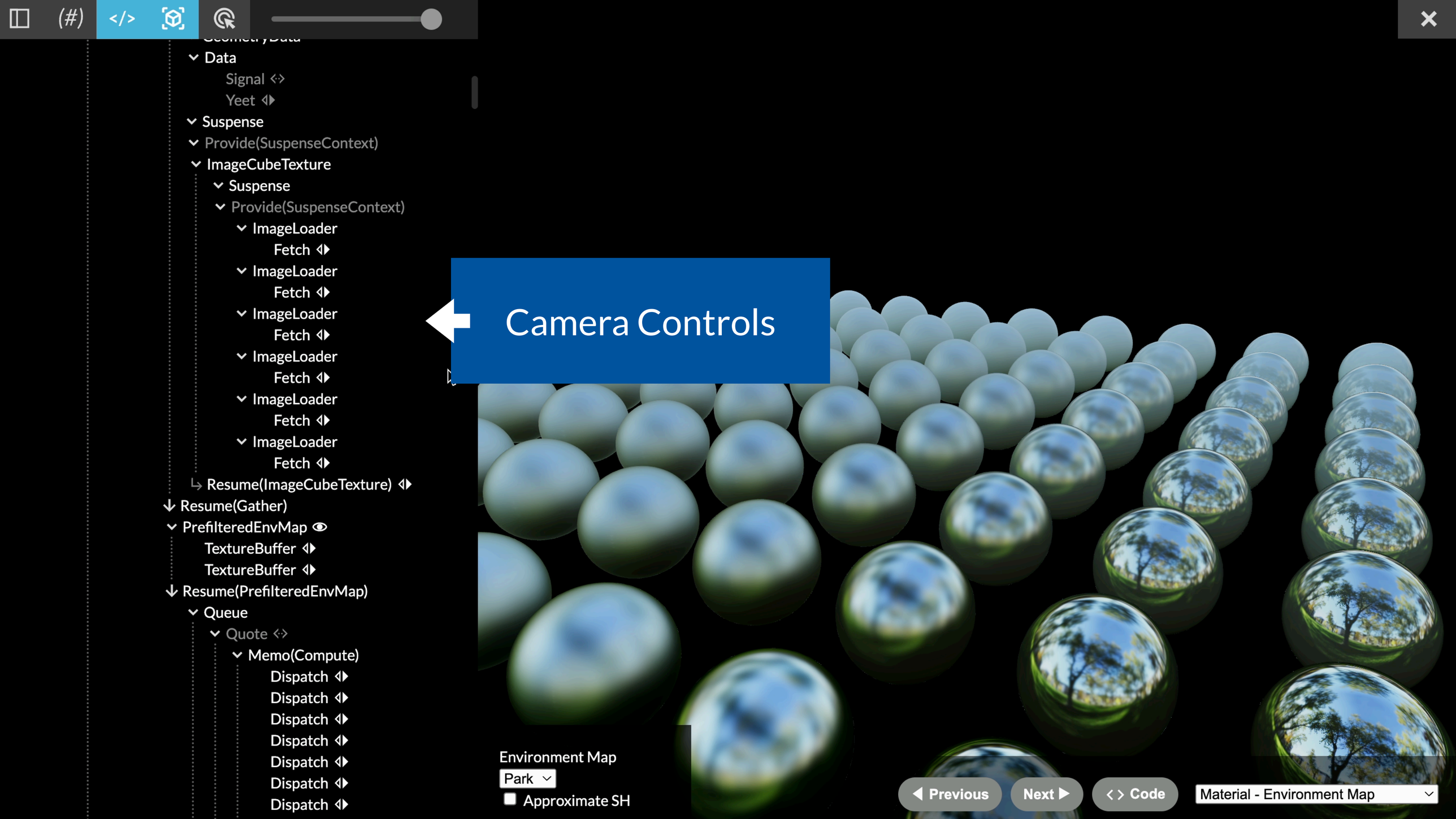

App
UseInspect
Memo(DebugProvider)
Provide(DebugContext)
WebGPU
Provide(DeviceContext)
Queue
Quote
AutoCanvas
AutoSize
Canvas
Signal
Provide(RenderContext)
Provide(LayoutContext)
PickingTarget
Provide(PickingContext)
Memo(DOMEvents)
Memo(EventProvider)
Provide(MouseContext)
Provide(WheelContext)
Provide(KeyboardContext)
Provide(EventContext)
CursorProvider
Capture(CursorState)
FontLoader
Fetch
Fetch
Fetch
Yeet
Resume(FontLoader)
FontProvider
Provide(FontContext)
Memo(Router)
Provide(RouterContext)
Memo(Routes)
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

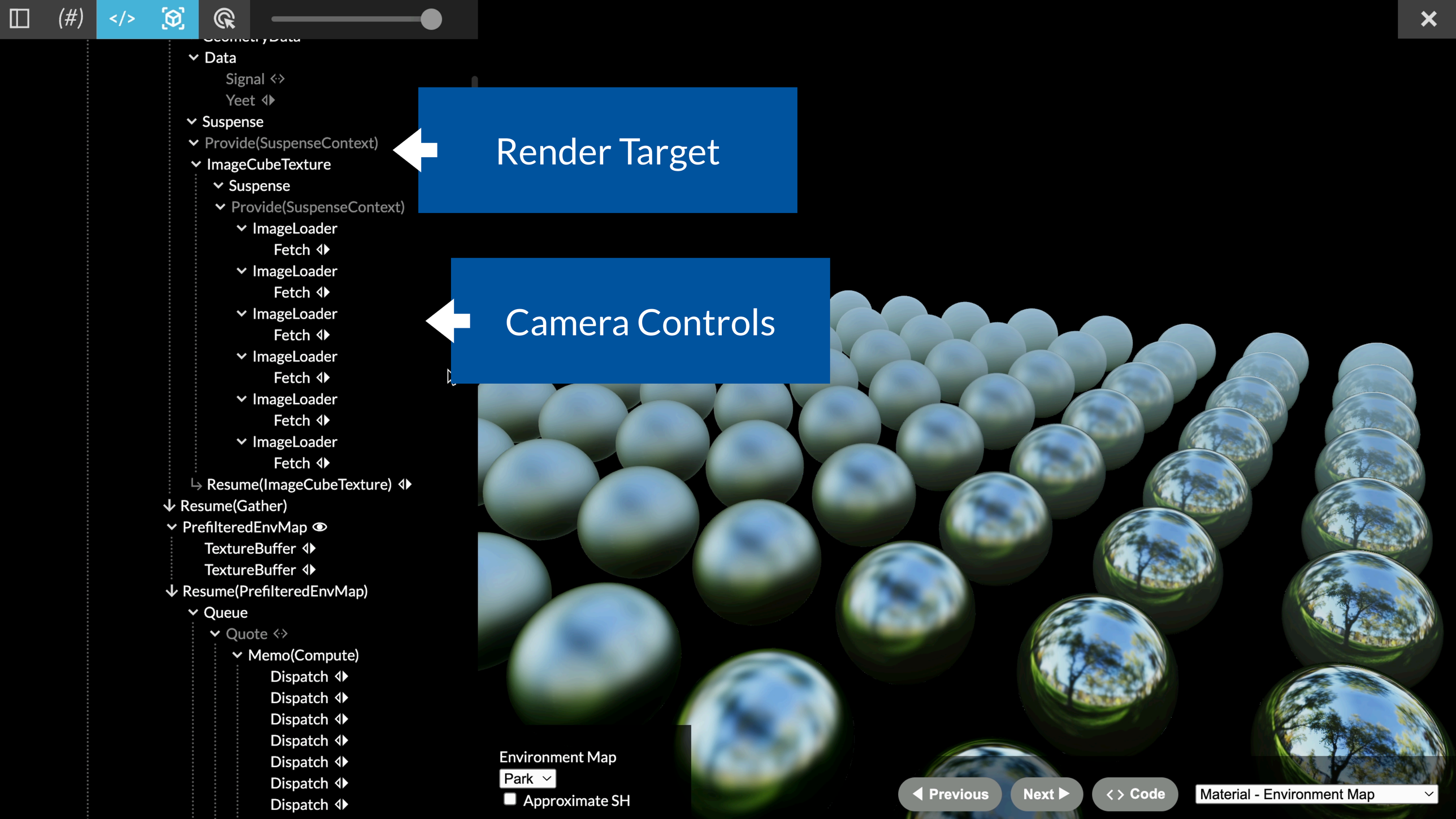

App
UseInspect
Memo(DebugProvider)
Provide(DebugContext)
WebGPU
Provide(DeviceContext)
Queue
Quote
AutoCanvas
AutoSize
Canvas
Signal
Provide(RenderContext)
Provide(LayoutContext)
PickingTarget
Provide(PickingContext)
Memo(DOMEvents)
Memo(EventProvider)
Provide(MouseContext)
Provide(WheelContext)
Provide(KeyboardContext)
Provide(EventContext)
CursorProvider
Capture(CursorState)
FontLoader
Fetch
Fetch
Fetch
Yeet
Resume(FontLoader)
FontProvider
Provide(FontContext)
Memo(Router)
Provide(RouterContext)
Memo(Routes)
Cube Map Loader
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

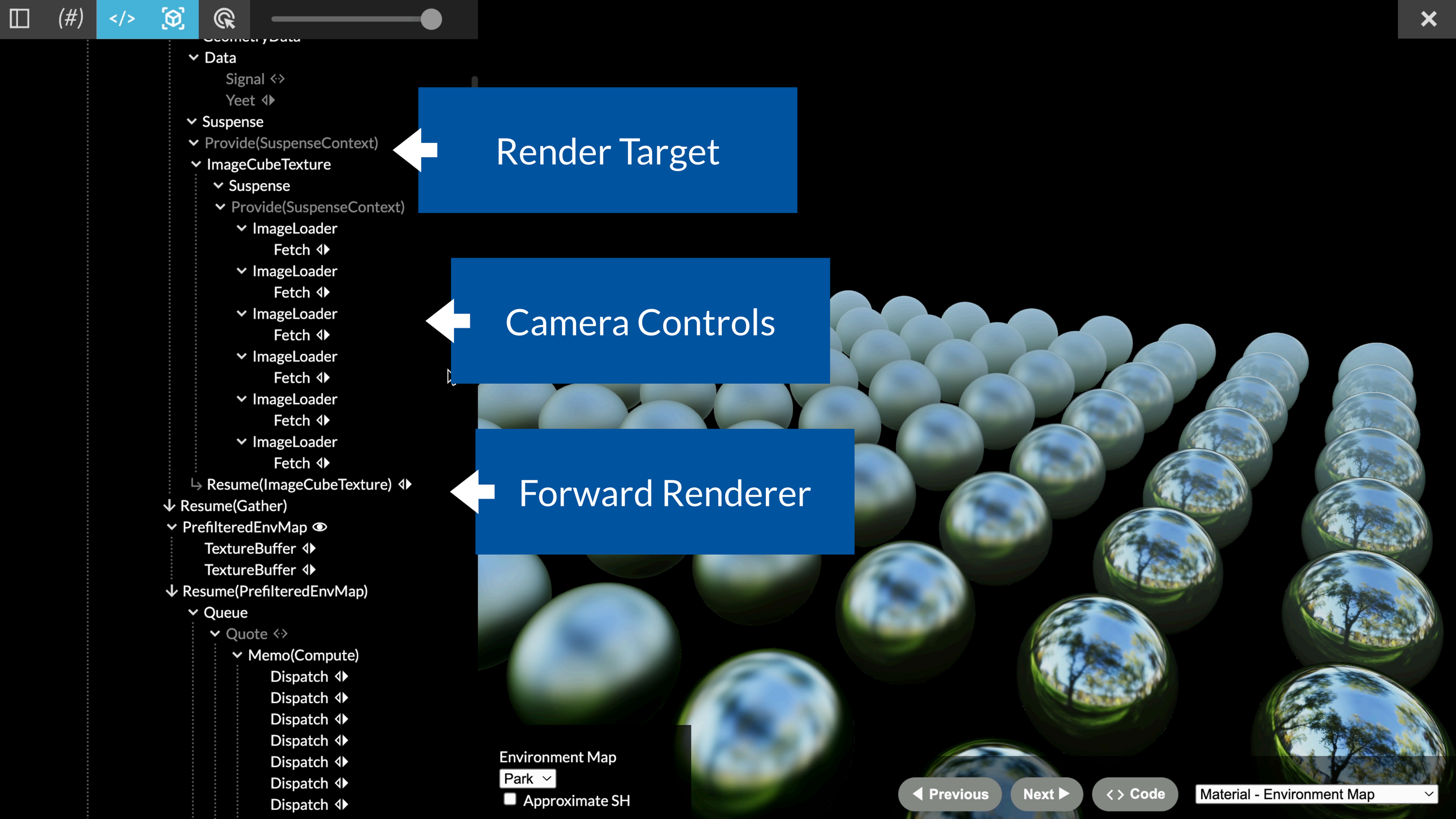

App
UseInspect
Memo(DebugProvider)
Provide(DebugContext)
WebGPU
Provide(DeviceContext)
Queue
Quote
AutoCanvas
AutoSize
Canvas
Signal
Provide(RenderContext)
Provide(LayoutContext)
PickingTarget
Provide(PickingContext)
Memo(DOMEvents)
Memo(EventProvider)
Provide(MouseContext)
Provide(WheelContext)
Provide(KeyboardContext)
Provide(EventContext)
CursorProvider
Capture(CursorState)
FontLoader
Fetch
Fetch
Fetch
Yeet
Resume(FontLoader)
FontProvider
Provide(FontContext)
Memo(Router)
Provide(RouterContext)
Memo(Routes)
Cube Map Loader
Compute Shaders
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

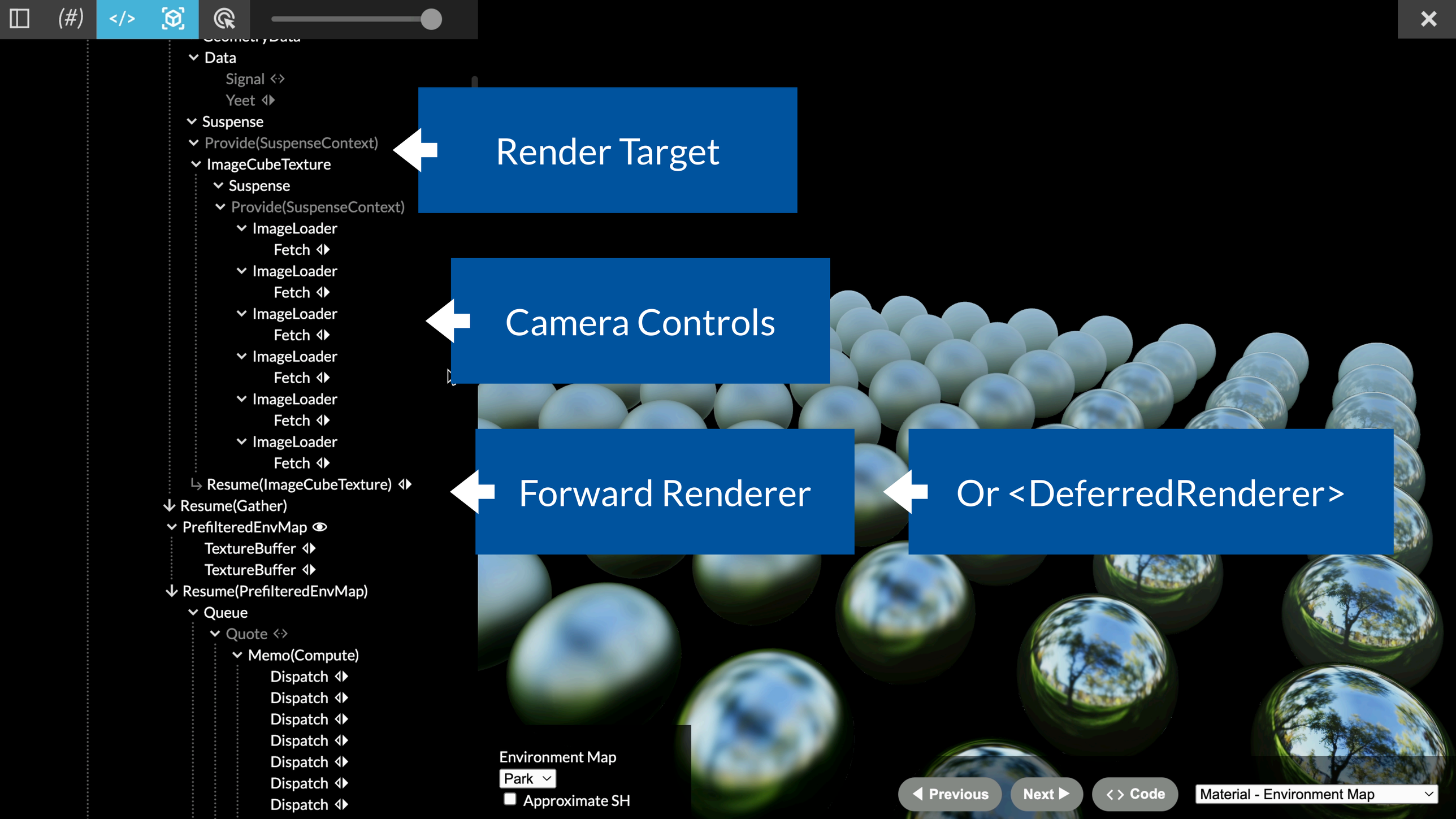

Data
Signal
Yeet
Suspense
Provide(SuspenseContext)
ImageCubeTexture
Suspense
Provide(SuspenseContext)
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
Resume(ImageCubeTexture)
Resume(Gather)
PrefilteredEnvMap
TextureBuffer
TextureBuffer
Resume(PrefilteredEnvMap)
Queue
Quote
Memo(Compute)
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

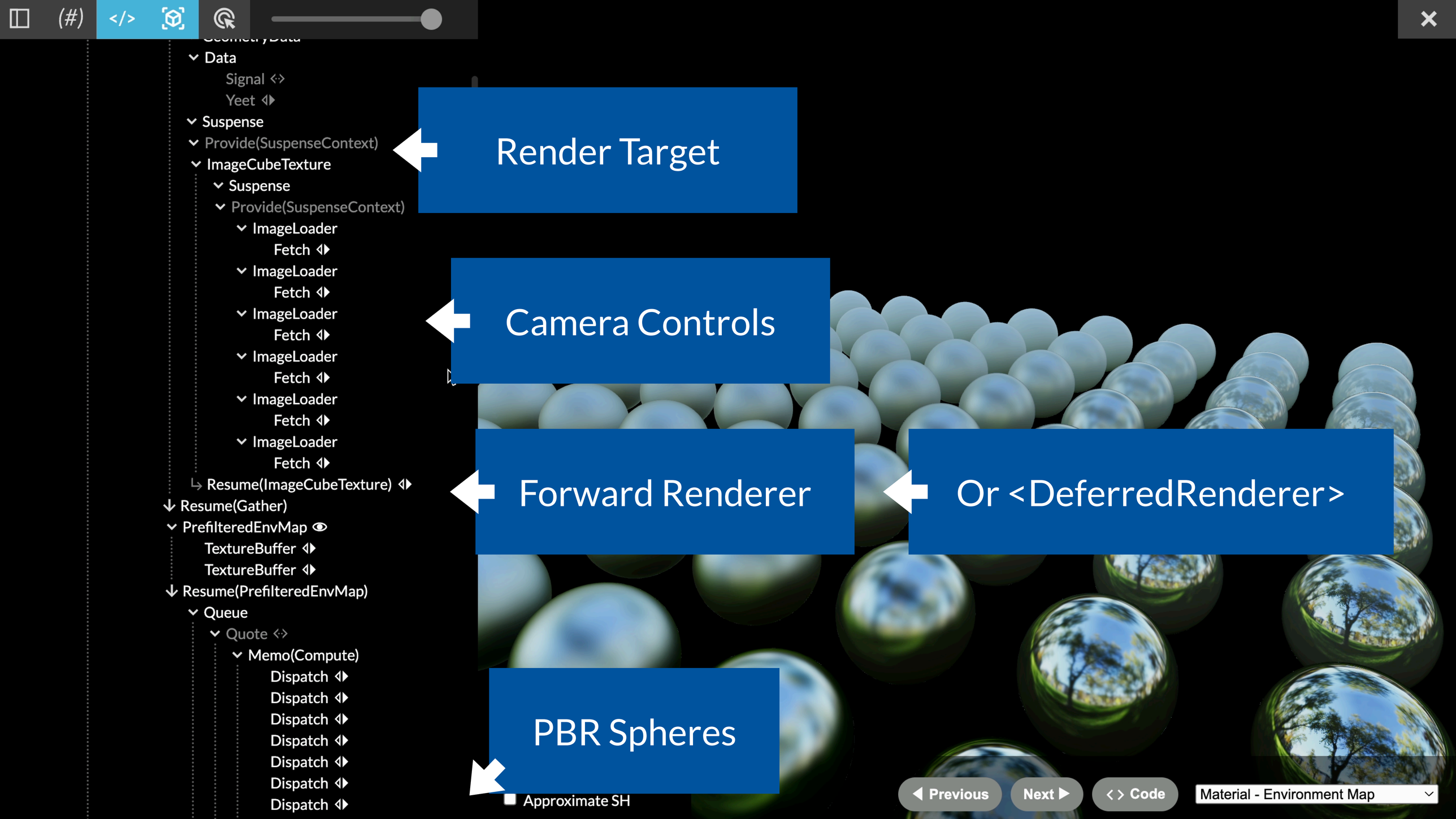

Data
Signal
Yeet
Suspense
Provide(SuspenseContext)
ImageCubeTexture
Suspense
Provide(SuspenseContext)
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
Resume(ImageCubeTexture)
Resume(Gather)
PrefilteredEnvMap
TextureBuffer
TextureBuffer
Resume(PrefilteredEnvMap)
Queue
Quote
Memo(Compute)
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

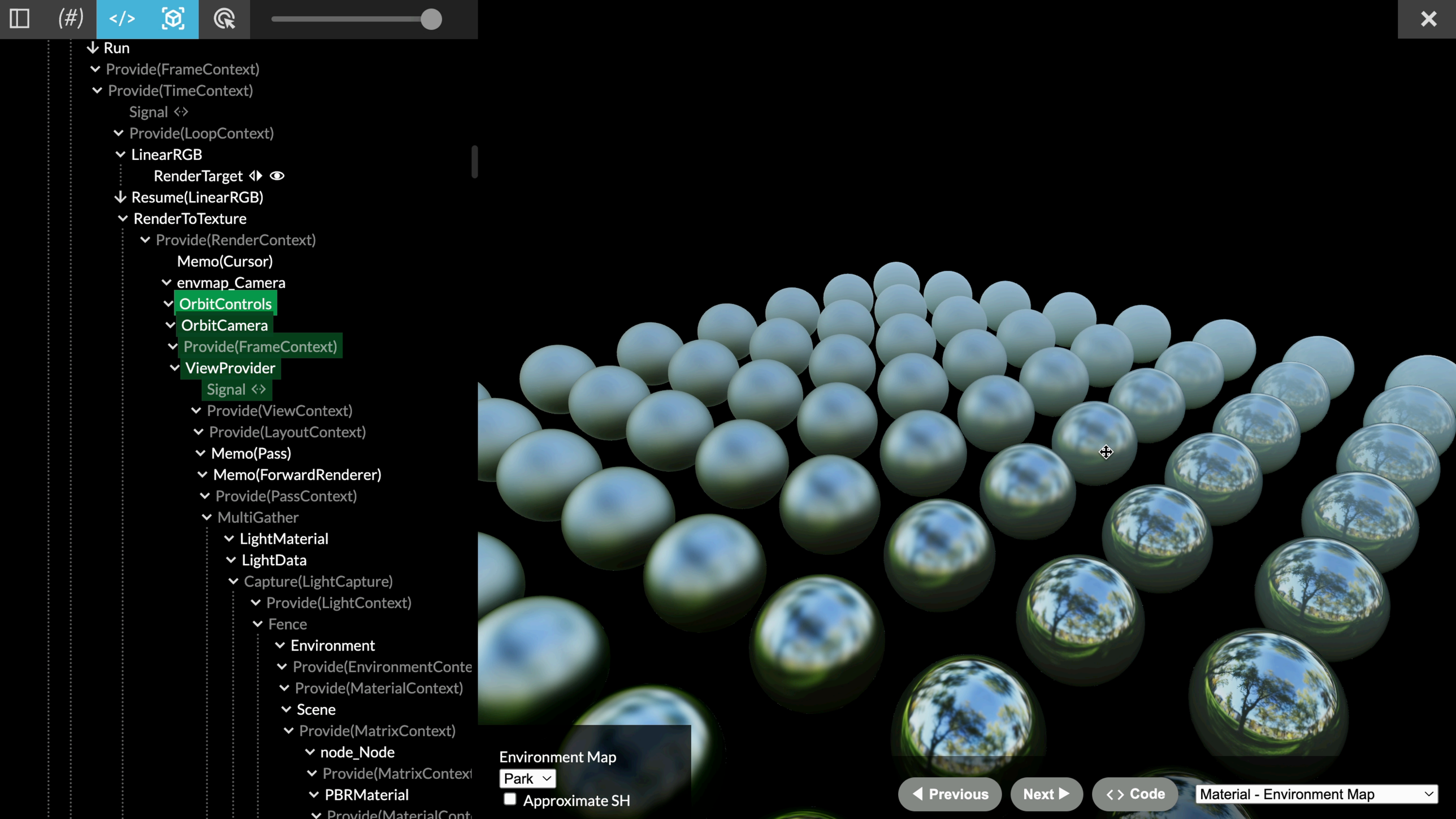

Data
Signal
Yeet
Suspense
Provide(SuspenseContext)
ImageCubeTexture
Suspense
Provide(SuspenseContext)
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
Resume(ImageCubeTexture)
Resume(Gather)
PrefilteredEnvMap
TextureBuffer
TextureBuffer
Resume(PrefilteredEnvMap)
Queue
Quote
Memo(Compute)
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Camera Controls
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

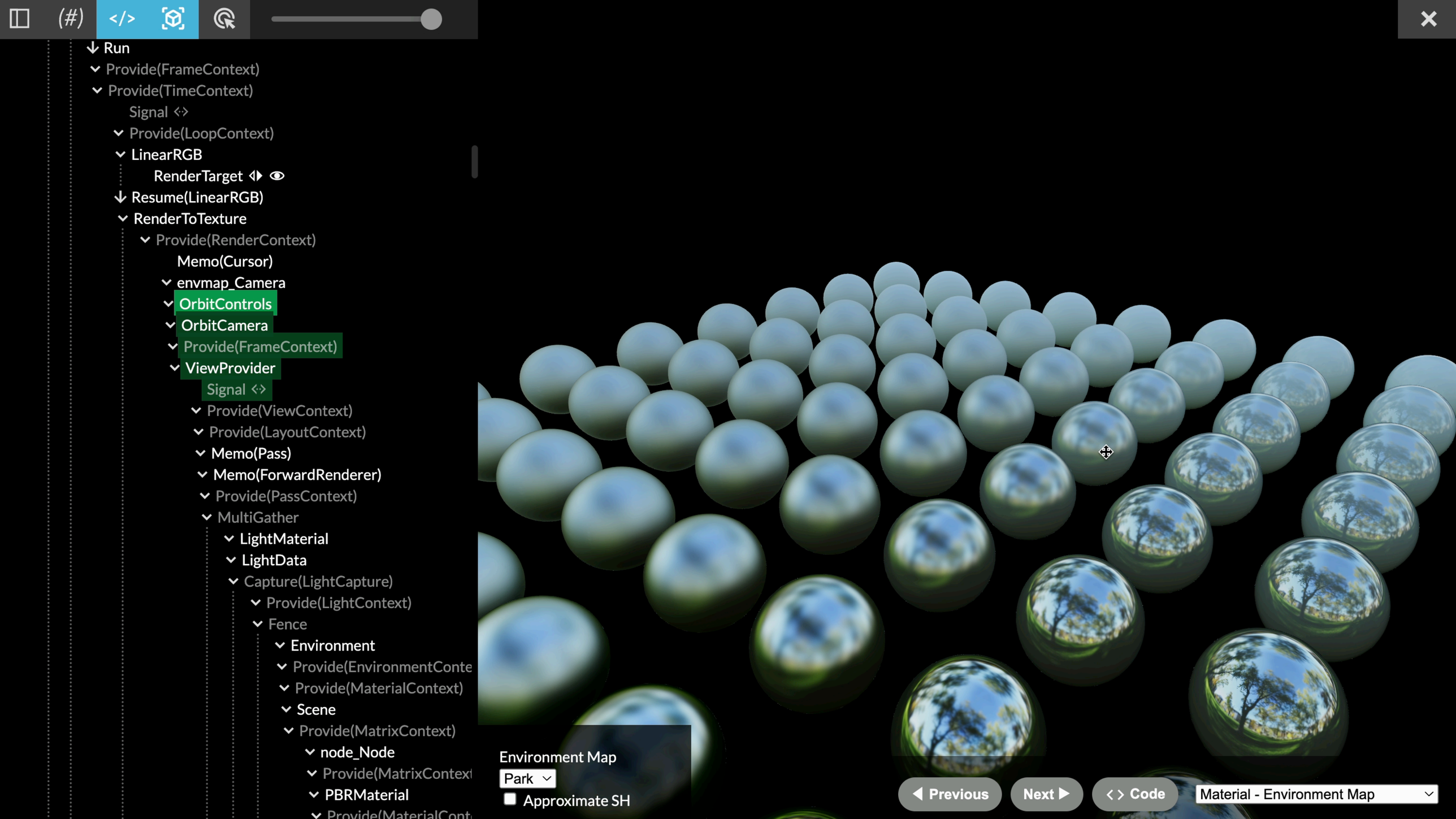

Data
Signal
Yeet
Suspense
Provide(SuspenseContext)
ImageCubeTexture
Suspense
Provide(SuspenseContext)
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
Resume(ImageCubeTexture)
Resume(Gather)
PrefilteredEnvMap
TextureBuffer
TextureBuffer
Resume(PrefilteredEnvMap)
Queue
Quote
Memo(Compute)
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Render Target
Camera Controls
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

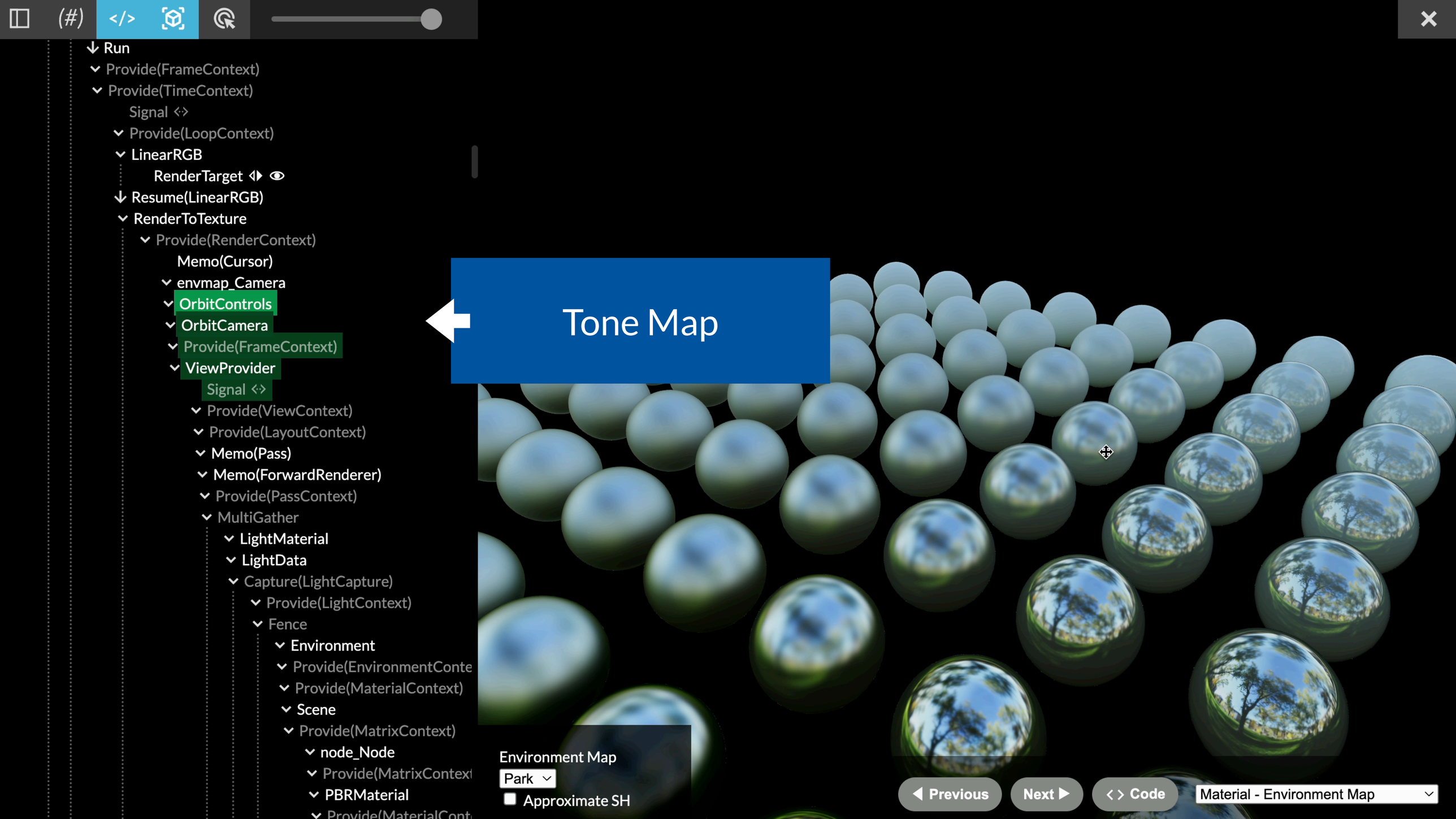

Data
Signal
Yeet
Suspense
Provide(SuspenseContext)
ImageCubeTexture
Suspense
Provide(SuspenseContext)
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
Resume(ImageCubeTexture)
Resume(Gather)
PrefilteredEnvMap
TextureBuffer
TextureBuffer
Resume(PrefilteredEnvMap)
Queue
Quote
Memo(Compute)
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Render Target
Camera Controls
Forward Renderer
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

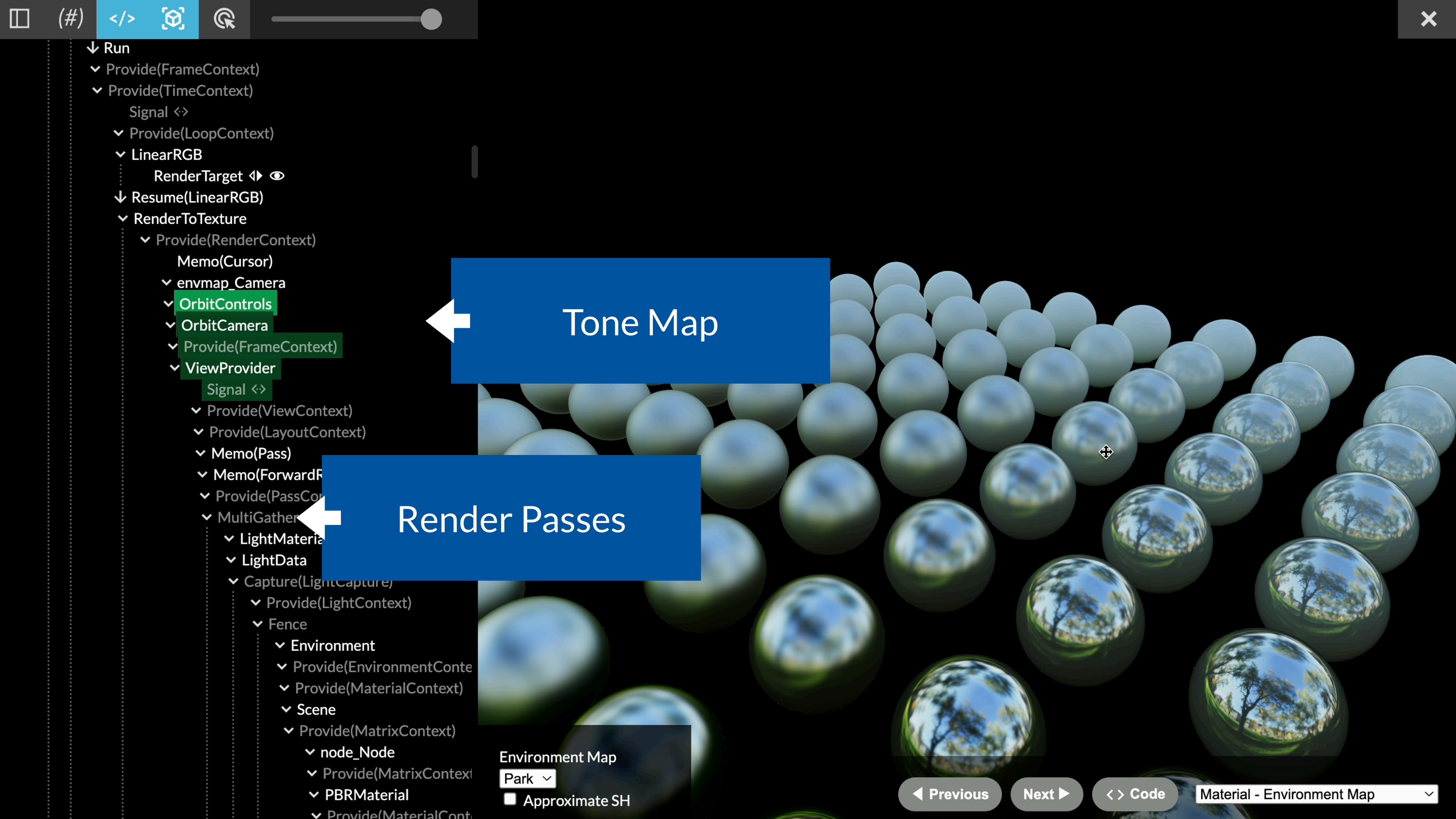

Data
Signal
Yeet
Suspense
Provide(SuspenseContext)
ImageCubeTexture
Suspense
Provide(SuspenseContext)
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
Resume(ImageCubeTexture)
Resume(Gather)
PrefilteredEnvMap
TextureBuffer
TextureBuffer
Resume(PrefilteredEnvMap)
Queue
Quote
Memo(Compute)
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Render Target
Camera Controls
Forward Renderer
Or <DeferredRenderer>
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

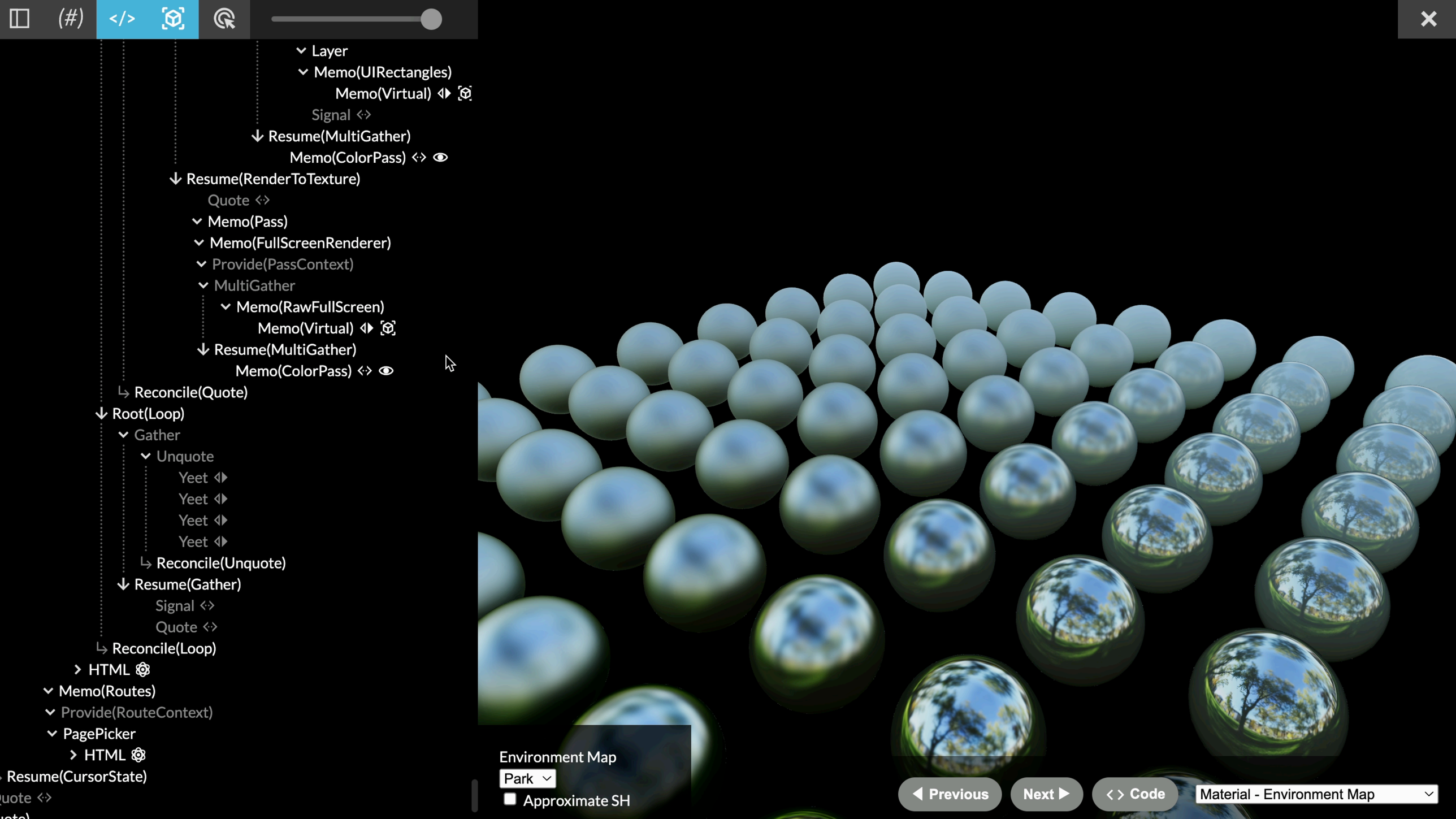

Data
Signal
Yeet
Suspense
Provide(SuspenseContext)
ImageCubeTexture
Suspense
Provide(SuspenseContext)
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
ImageLoader
Fetch
Resume(ImageCubeTexture)
Resume(Gather)
PrefilteredEnvMap
TextureBuffer
TextureBuffer
Resume(PrefilteredEnvMap)
Queue
Quote
Memo(Compute)
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Dispatch
Render Target
Camera Controls
Forward Renderer
Or <DeferredRenderer>
PBR Spheres
Approximate SH
Previous
Next
Code
Material - Environment Map

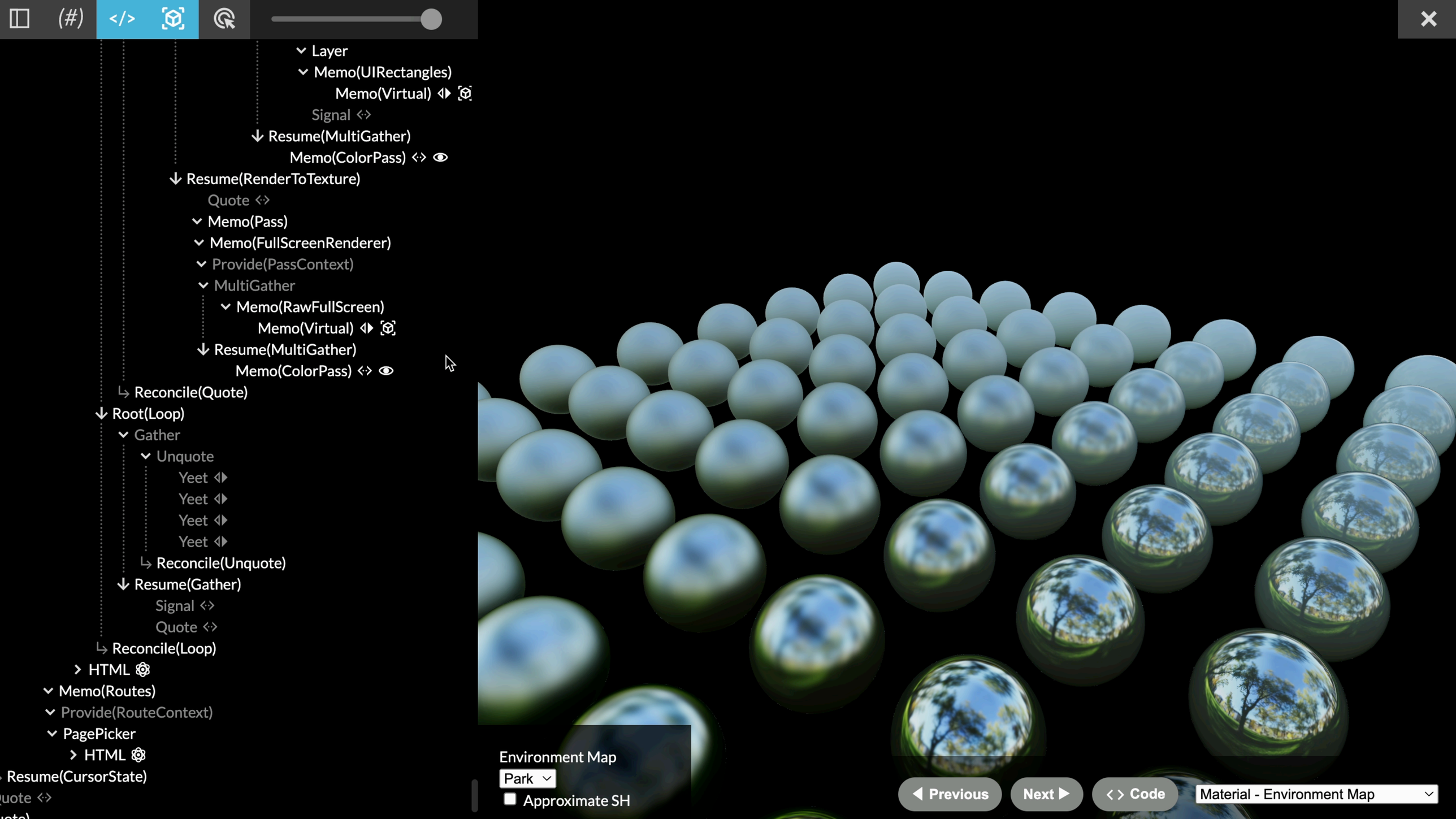

Run
Provide(FrameContext)
Provide(TimeContext)
Signal
Provide(LoopContext)
LinearRGB
RenderTarget
Resume(LinearRGB)
RenderToTexture
Provide(RenderContext)
Memo(Cursor)
envmap_Camera
OrbitControls
OrbitCamera
Provide(FrameContext)
ViewProvider
Signal
Provide(ViewContext)
Provide(LayoutContext)
Memo(Pass)
Memo(ForwardRenderer)
Provide(PassContext)
MultiGather
LightMaterial
LightData
Capture(LightCapture)
Provide(LightContext)
Fence
Environment
Provide(EnvironmentConte
Provide(MaterialContext)
Scene
Provide(MatrixContext)
node_Node
Provide(MatrixContext
PBRMaterial
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

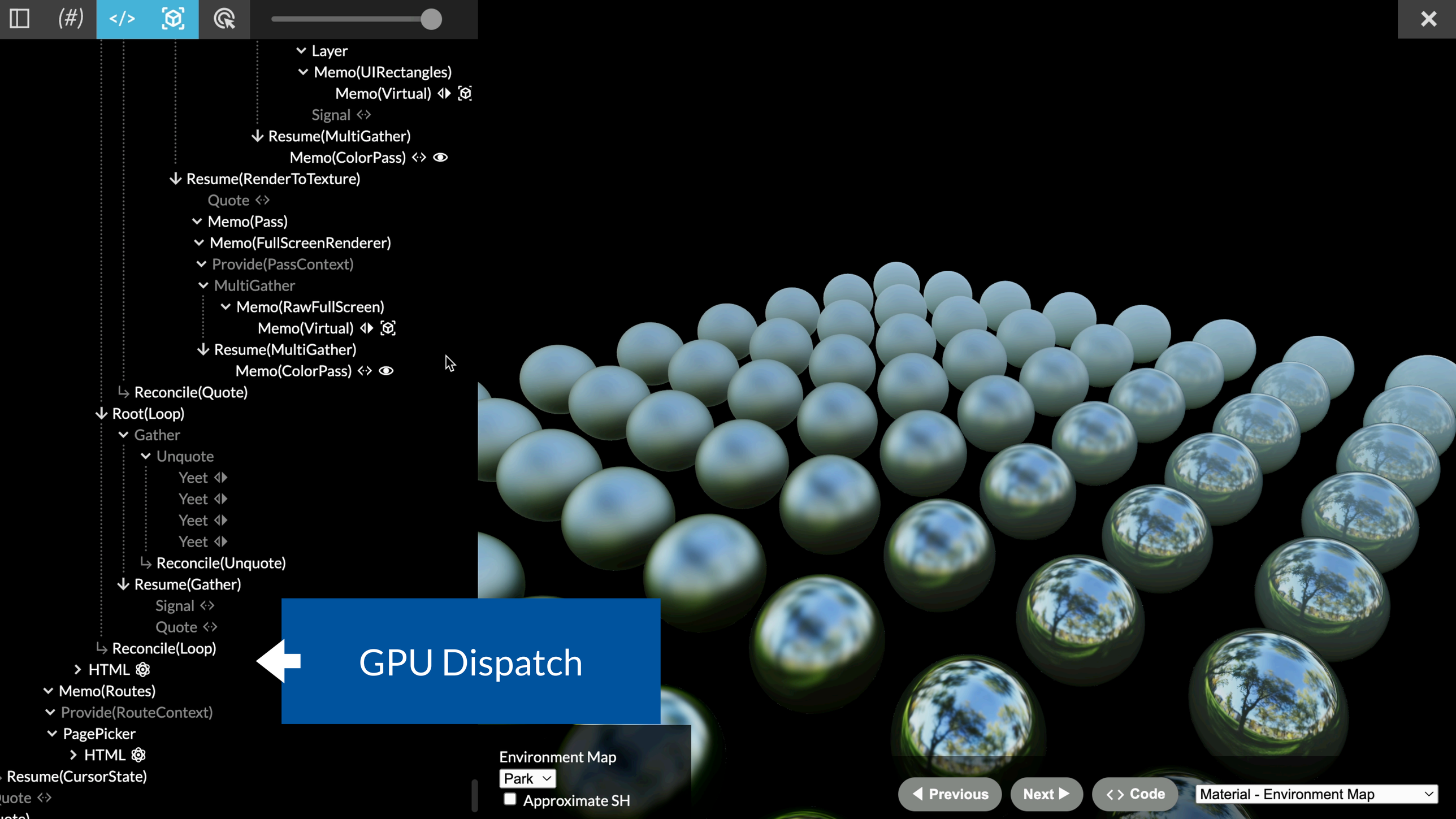

Run
Provide(FrameContext)
Provide(TimeContext)
Signal
Provide(LoopContext)
LinearRGB
RenderTarget
Resume(LinearRGB)
RenderToTexture
Provide(RenderContext)
Memo(Cursor)
envmap_Camera
OrbitControls
OrbitCamera
Provide(FrameContext)
ViewProvider
Signal
Provide(ViewContext)
Provide(LayoutContext)
Memo(Pass)
Memo(ForwardRenderer)
Provide(PassContext)
MultiGather
LightMaterial
LightData
Capture(LightCapture)
Provide(LightContext)
Fence
Environment
Provide(EnvironmentConte
Provide(MaterialContext)
Scene
Provide(MatrixContext)
node_Node
Provide(MatrixContext
PBRMaterial
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

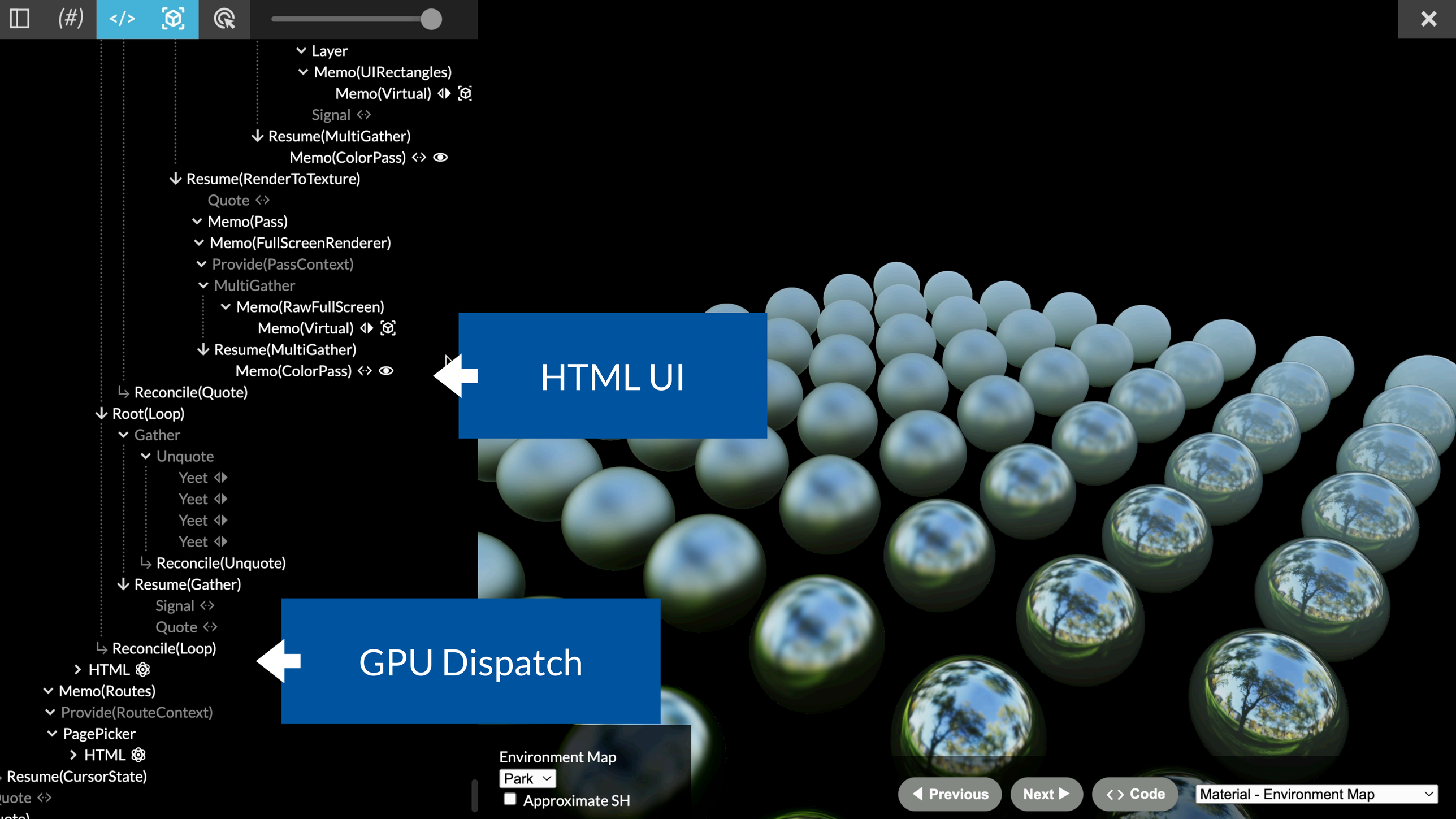

Run
Provide(FrameContext)
Provide(TimeContext)
Signal
Provide(LoopContext)
LinearRGB
RenderTarget
Resume(LinearRGB)
RenderToTexture
Provide(RenderContext)
Memo(Cursor)
envmap_Camera
OrbitControls
OrbitCamera
Provide(FrameContext)
ViewProvider
Signal
Provide(ViewContext)
Provide(LayoutContext)
Memo(Pass)
Memo(ForwardRenderer)
Provide(PassContext)
MultiGather
LightMaterial
LightData
Capture(LightCapture)
Provide(LightContext)
Fence
Environment
Provide(EnvironmentConte
Provide(MaterialContext)
Scene
Provide(MatrixContext)
node_Node
Provide(MatrixContext
PBRMaterial
Tone Map
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map


Run
Provide(FrameContext)
Provide(TimeContext)
Signal
Provide(LoopContext)
LinearRGB
RenderTarget
Resume(LinearRGB)
RenderToTexture
Provide(RenderContext)
Memo(Cursor)
envmap_Camera
OrbitControls
OrbitCamera
Provide(FrameContext)
ViewProvider
Signal
Provide(ViewContext)
Provide(LayoutContext)
Memo(Pass)
Memo(ForwardR
Provide(PassCo
MultiGather
LightMateria
LightData
Capture(LightCapture)
Provide(LightContext)
Fence
Environment
Provide(EnvironmentConte
Provide(MaterialContext)
Scene
Provide(MatrixContext)
node_Node
Provide(MatrixContext
PBRMaterial
Tone Map
Render Passes
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

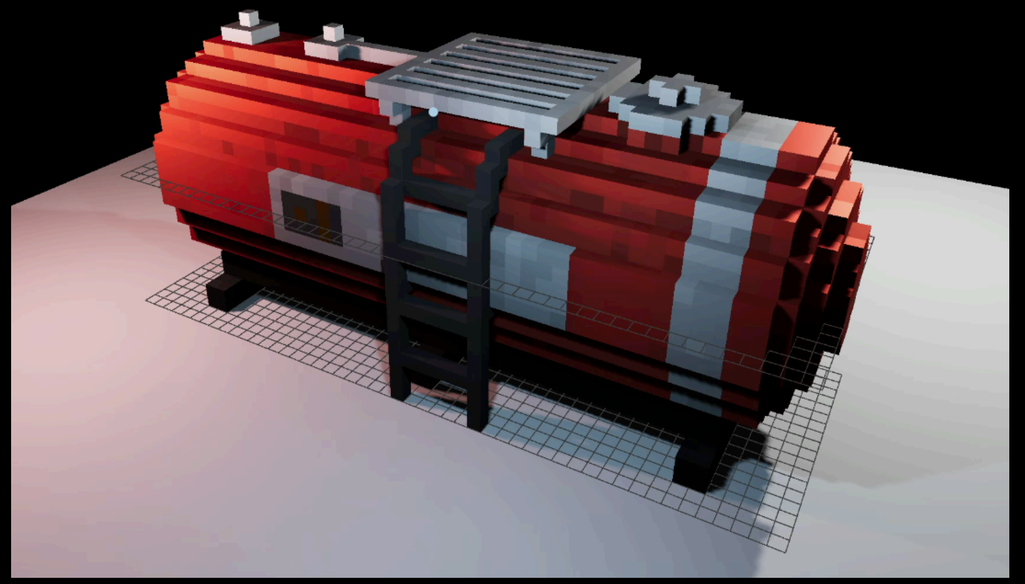

Layer
Memo(UIRectangles)
Memo(Virtual)
Signal
Resume(MultiGather)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Reconcile(Unquote)
Resume(Gather)
Signal
Quote
Reconcile(Loop)
HTML
Memo(Routes)
Provide(RouteContext)
PagePicker
HTML
Resume(CursorState)
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map


Layer
Memo(UIRectangles)
Memo(Virtual)
Signal
Resume(MultiGather)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Reconcile(Unquote)
Resume(Gather)
Signal
Quote
Reconcile(Loop)
HTML
Memo(Routes)
Provide(RouteContext)
PagePicker
HTML
Resume(CursorState)
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

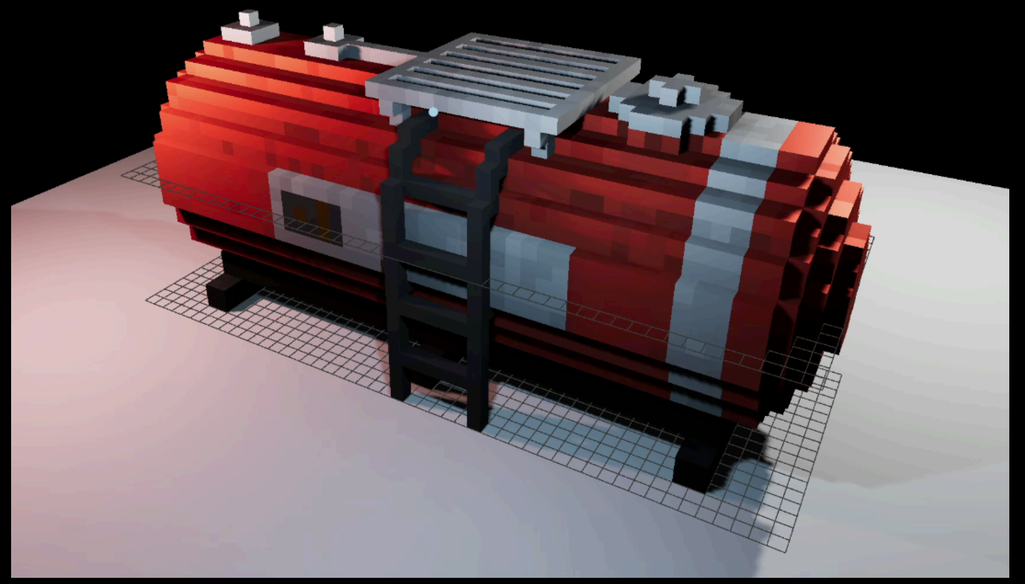

Layer
Memo(UIRectangles)
Memo(Virtual)
Signal
Resume(MultiGather)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Reconcile(Unquote)
Resume(Gather)
Signal
Quote
Reconcile(Loop)
HTML
Memo(Routes)
Provide(RouteContext)
PagePicker
HTML
Resume(CursorState)
GPU Dispatch
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map


Layer
Memo(UIRectangles)
Memo(Virtual)
Signal
Resume(MultiGather)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Reconcile(Unquote)
Resume(Gather)
Signal
Quote
Reconcile(Loop)
HTML
Memo(Routes)
Provide(RouteContext)
PagePicker
HTML
Resume(CursorState)
HTML UI
GPU Dispatch
Environment Map
Park
Approximate SH
Previous
Next
Code
Material - Environment Map

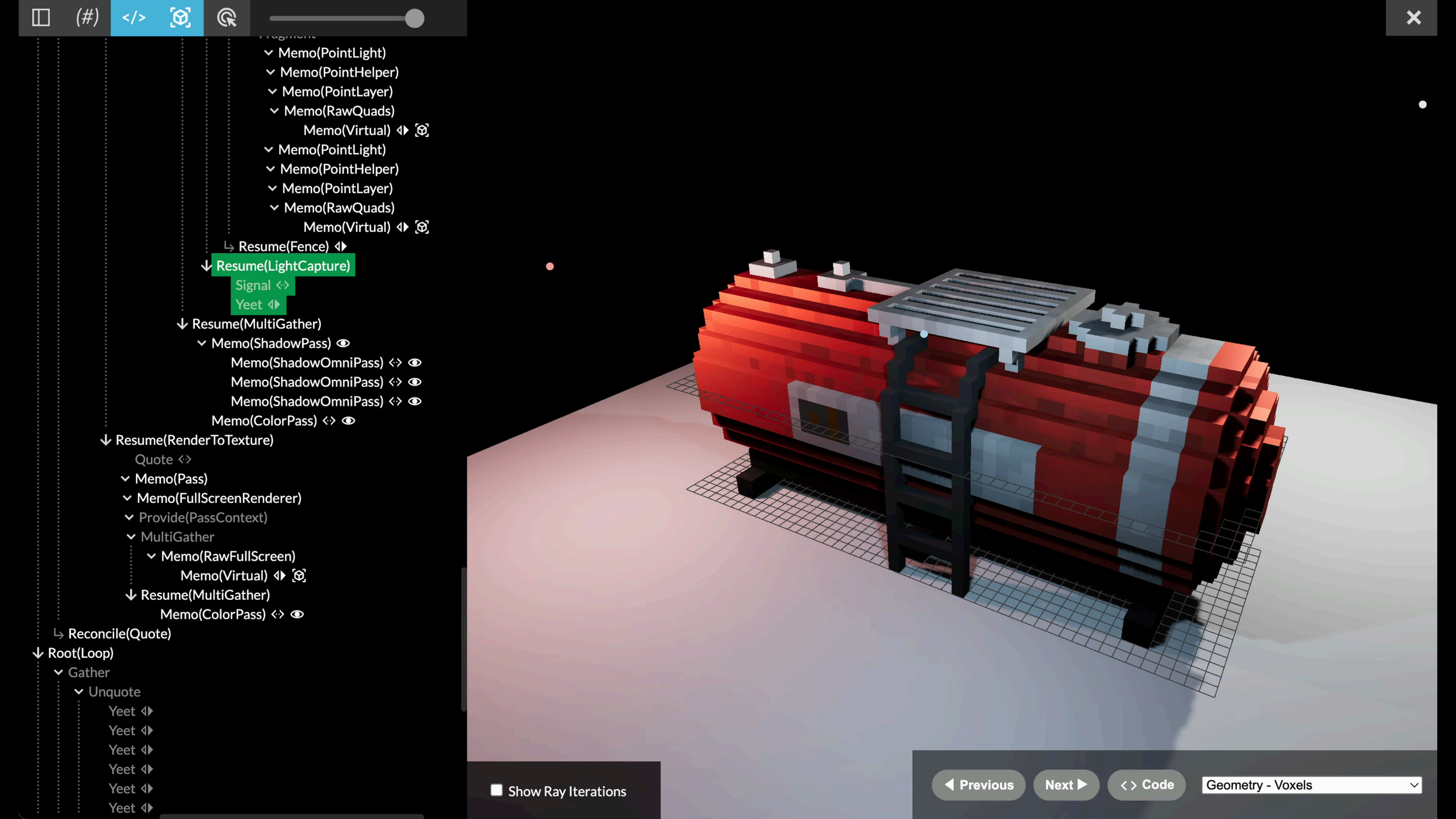
```
<LinearRGB tonemap="aces" gain={2} samples={1}>
  <Cursor cursor='move' />
  <Camera>
    <Pass lights shadows>
      <AmbientLight color={[1, 1, 1, 1]} intensity={0.01} />

      <Environment preset="none">
        <Scene>
          <Node rotation={[90, 180, 0]}>

            <VoxData url={url}>{
              (vox: Vox) => <VoxModel vox={vox} flat />
            }</VoxData>

            <Primitive>
              <Plot>
                <Cartesian
                  range={[[-9, 9], [-25, 25], [-10, 10]]}
                  scale={[9, 25, 10]}
                >
                  <Grid
                    origin={[0, 0, -11]}
                    axes='xy'
                    width={2}
                    first={{ detail: 3, divide: 18, end: true }}
                    second={{ detail: 3, divide: 48, end: true }}
                    depth={0.5}
                    zBias={1}
```

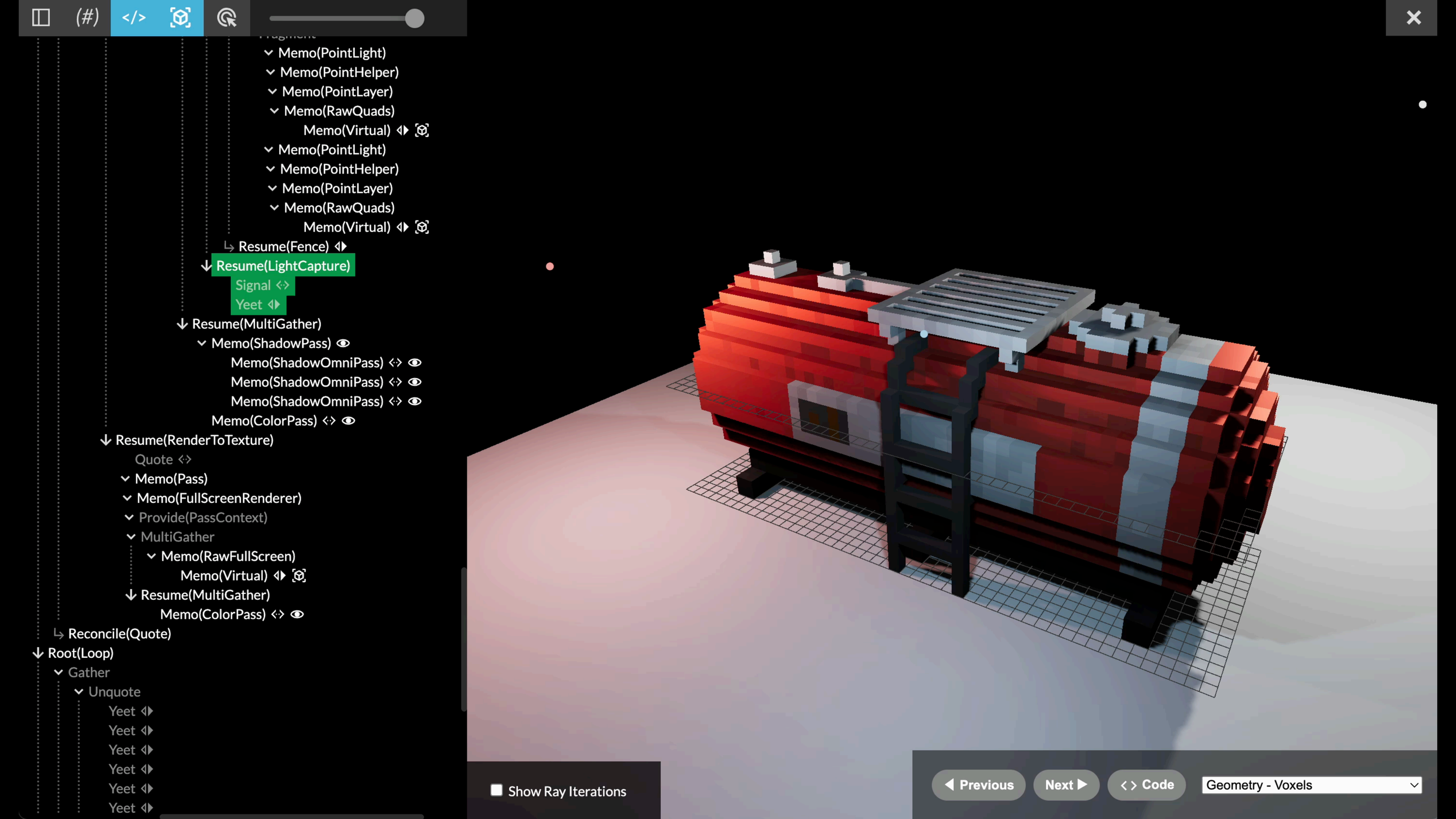


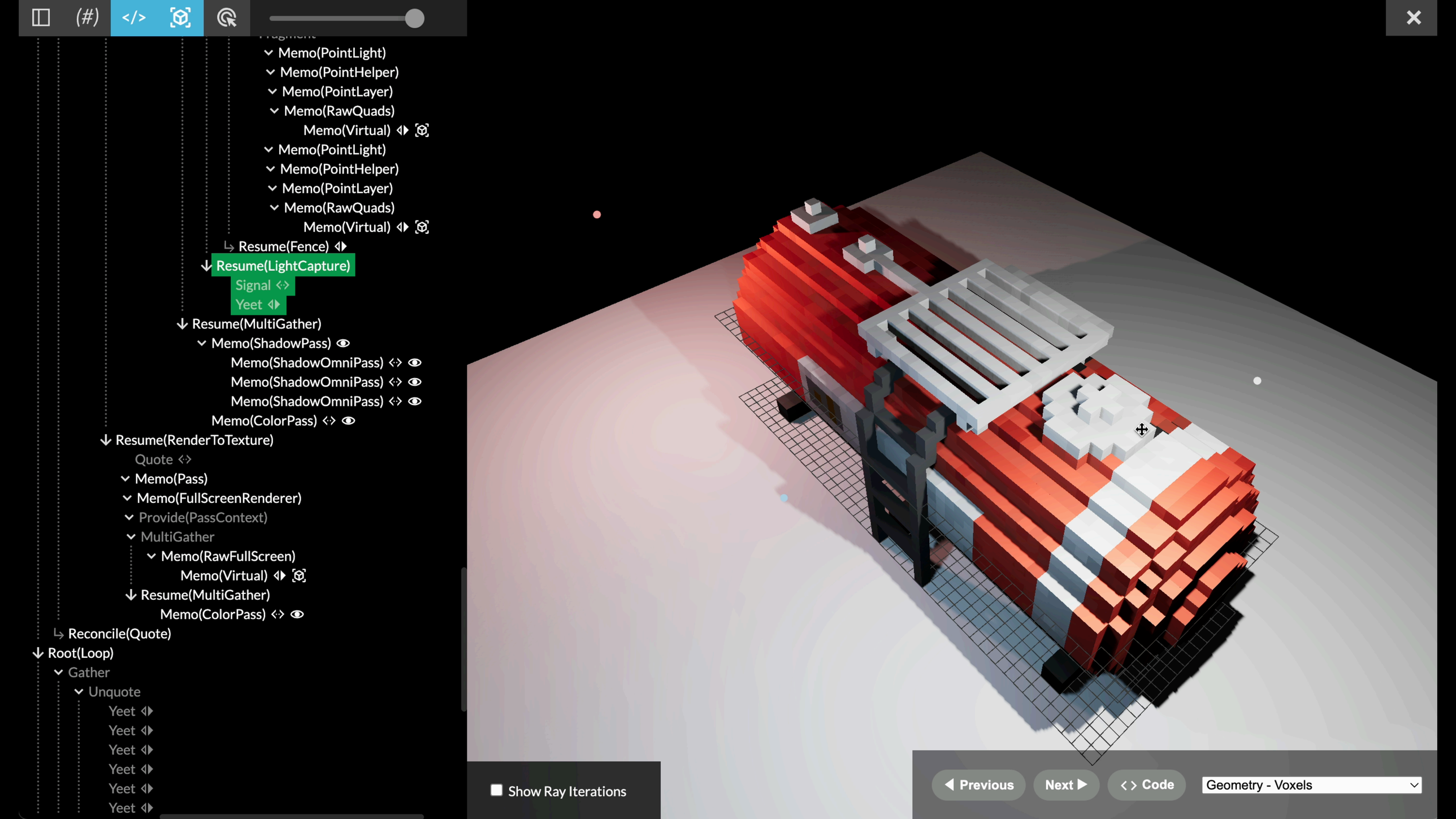
```
<LinearRGB tonemap="aces" gain={2} samples={1}>
  <Cursor cursor='move' />
  <Camera>
    <Pass lights shadows>
      <AmbientLight color={[1, 1, 1, 1]} intensity={0.01} />

      <Environment preset="none">
        <Scene>
          <Node rotation={[90, 180, 0]}>

            <VoxData url={url}>{
              (vox: Vox) => <VoxModel vox={vox} flat />
            }</VoxData>

            <Primitive>
              <Plot>
                <Cartesian
                  range={[[-9, 9], [-25, 25], [-10, 10]]}
                  scale={[9, 25, 10]}
                >
                  <Grid
                    origin={[0, 0, -11]}
                    axes='xy'
                    width={2}
                    first={{ detail: 3, divide: 18, end: true }}
                    second={{ detail: 3, divide: 48, end: true }}
                    depth={0.5}
                    zBias={1}
```

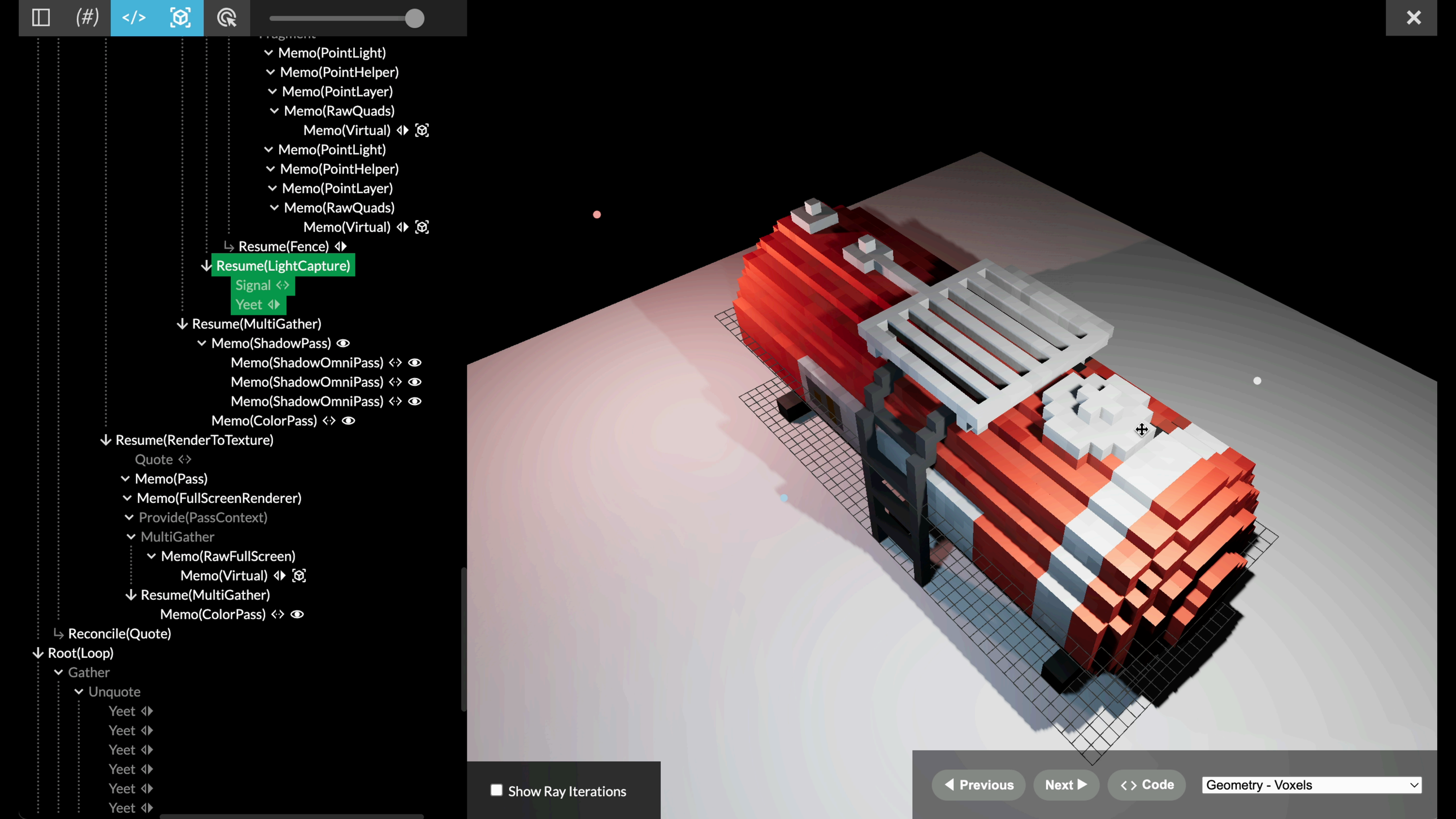


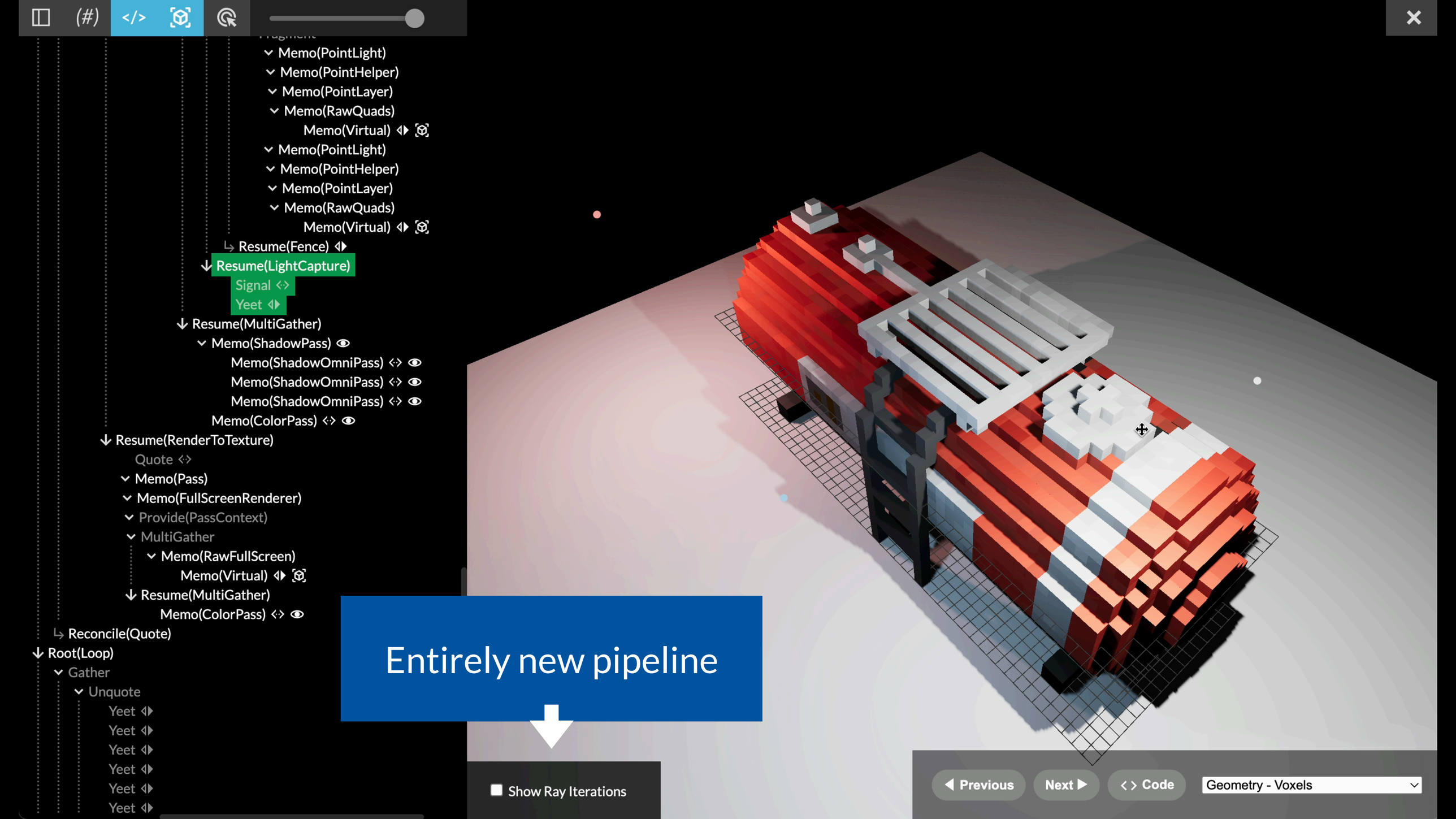
@use-gpu/voxel

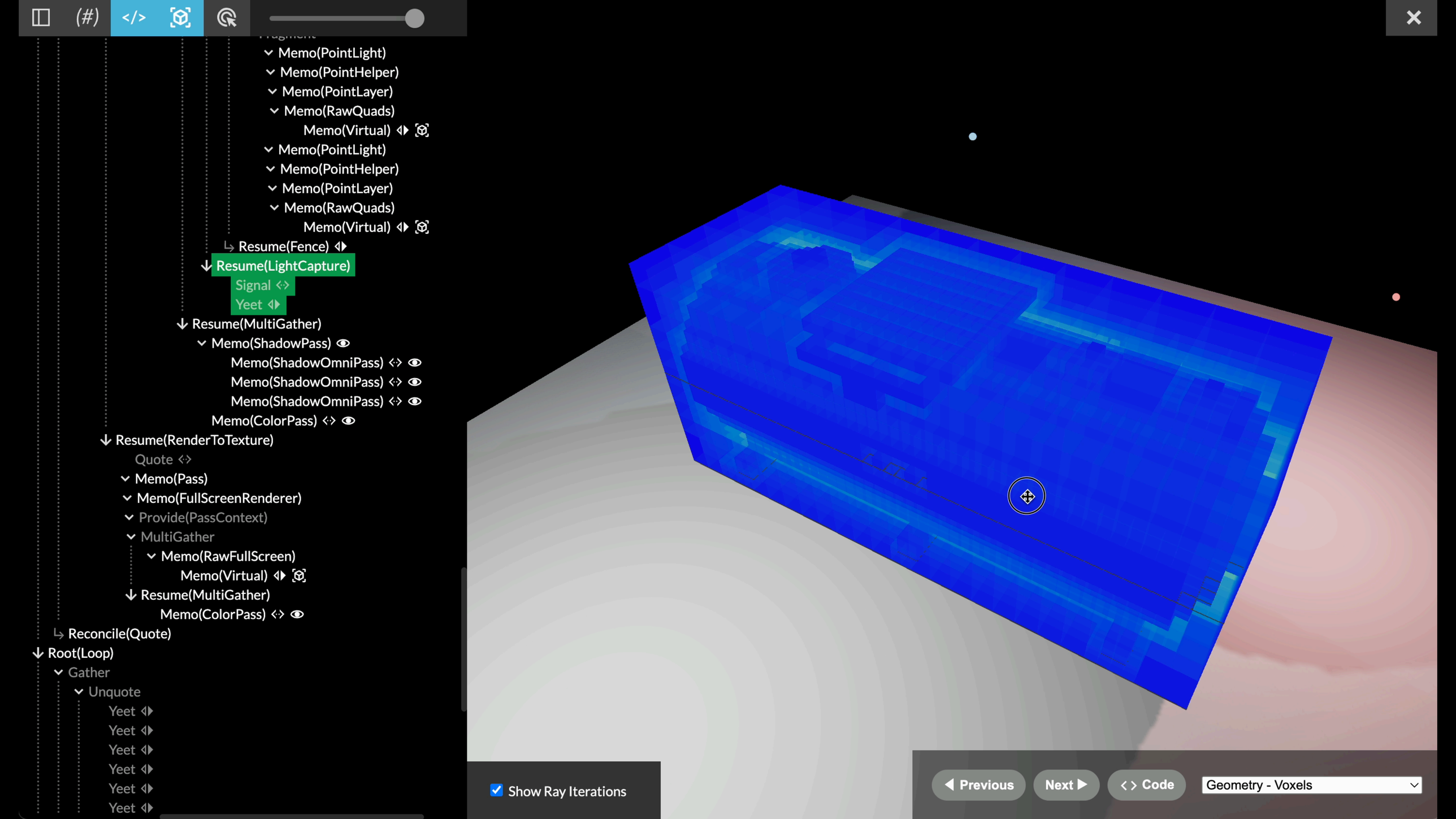

Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels

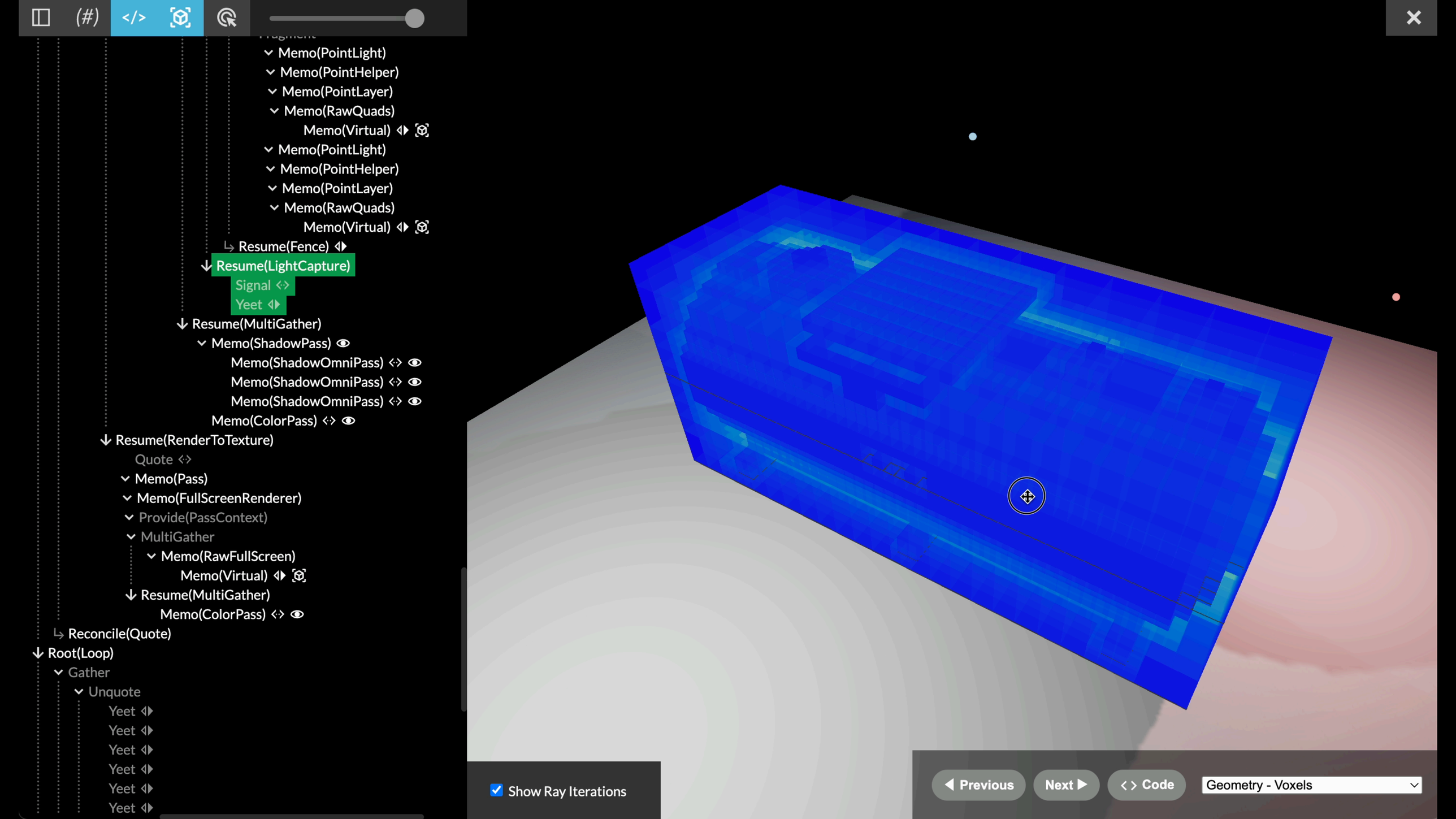

Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels


Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels


Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels

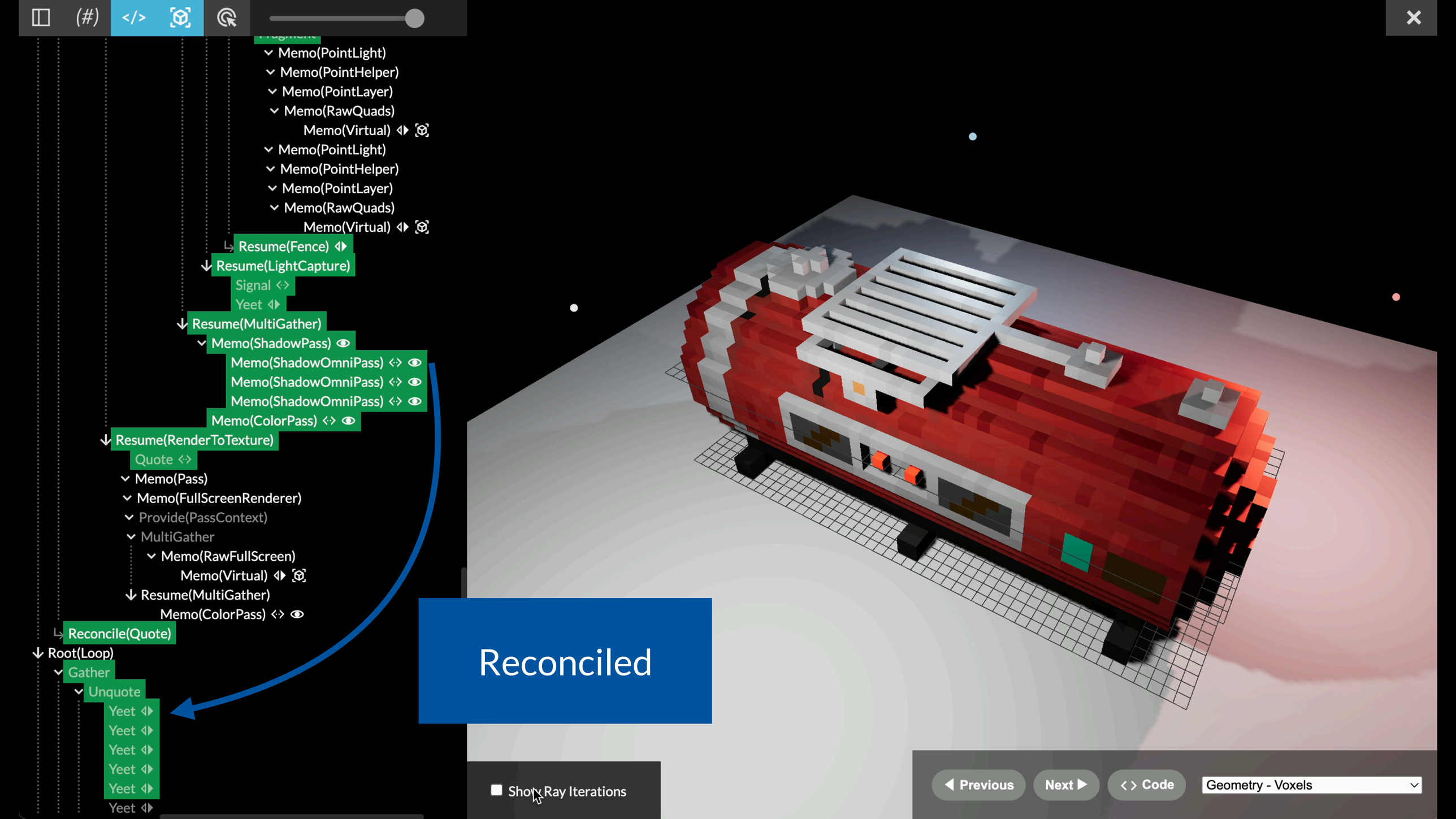

Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Entirely new pipeline
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels

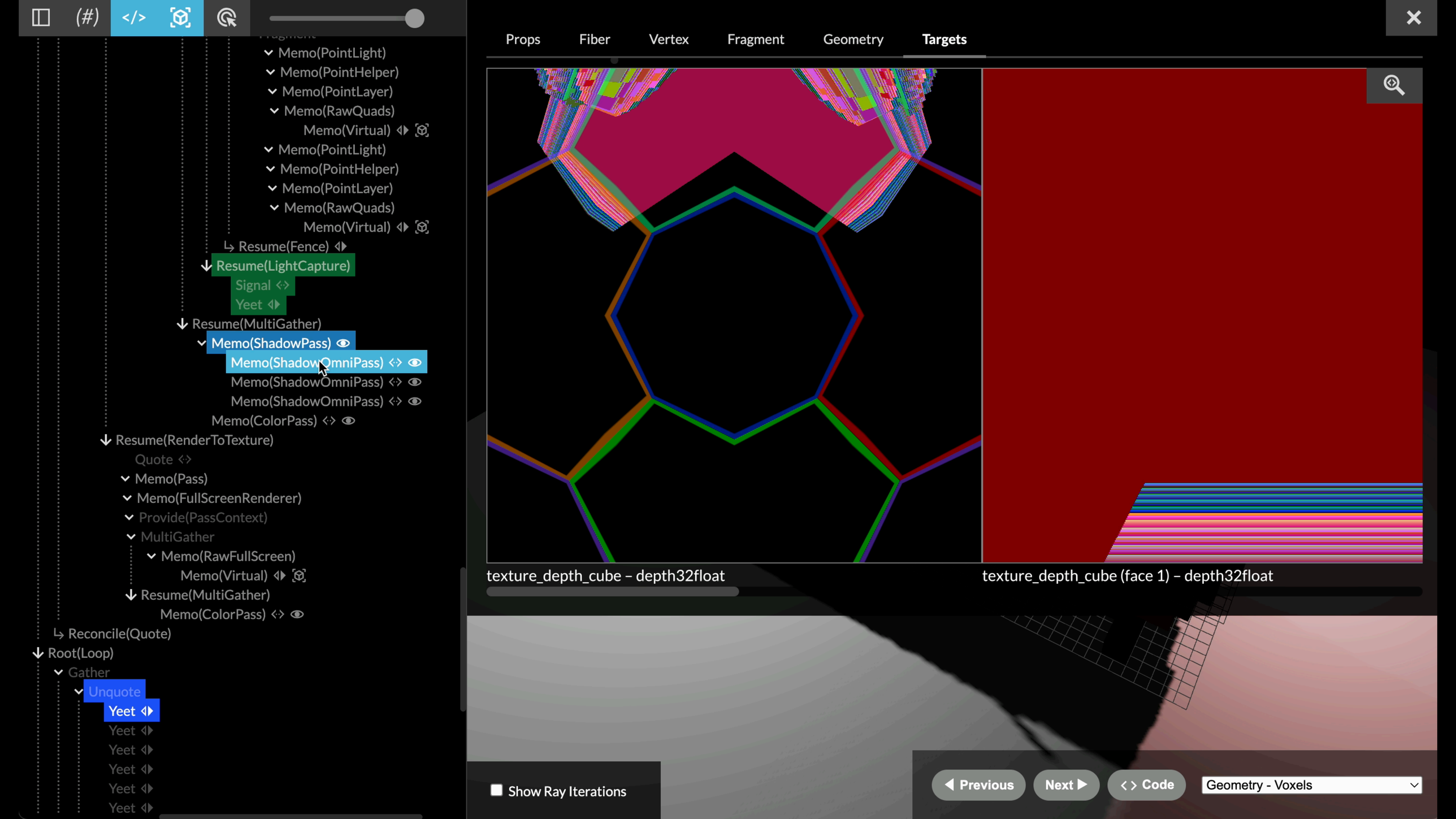

Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels

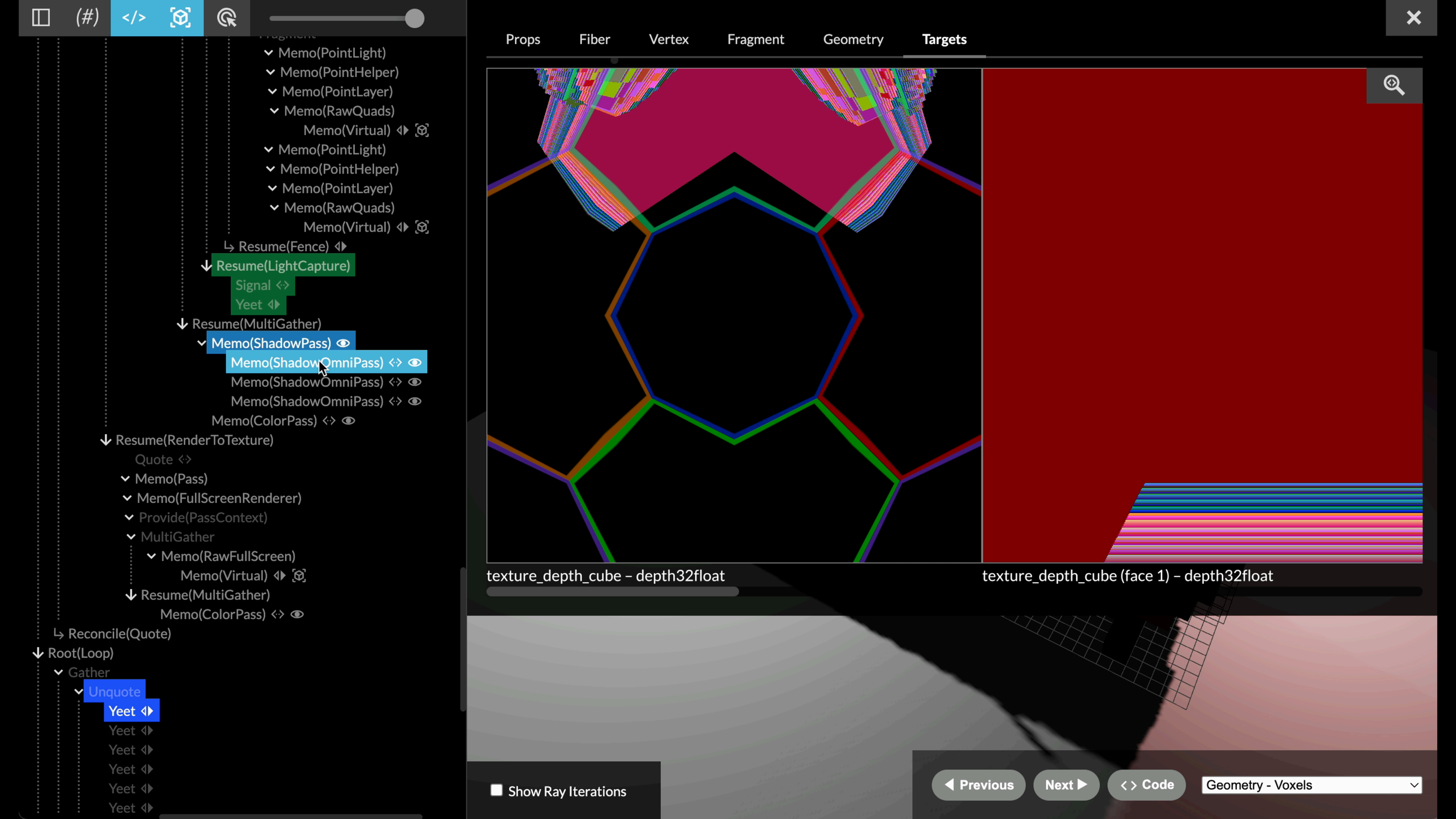

Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels


Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels


Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels


Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Reconciled
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels


Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Props
Fiber
Vertex
Fragment
Geometry
Targets
texture_depth_cube – depth32float
texture_depth_cube (face 1) – depth32float
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels


Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Props
Fiber
Vertex
Fragment
Geometry
Targets
texture_depth_cube – depth32float
texture_depth_cube (face 1) – depth32float
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels

Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet

Props Fiber Vertex Fragment Geometry **Targets**

texture_depth_cube (face 5) – depth32float

texture_depth_cube (face 6) – depth32float

Show Ray Iterations

◀ Previous Next ▶ < > Code Geometry - Voxels

Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet

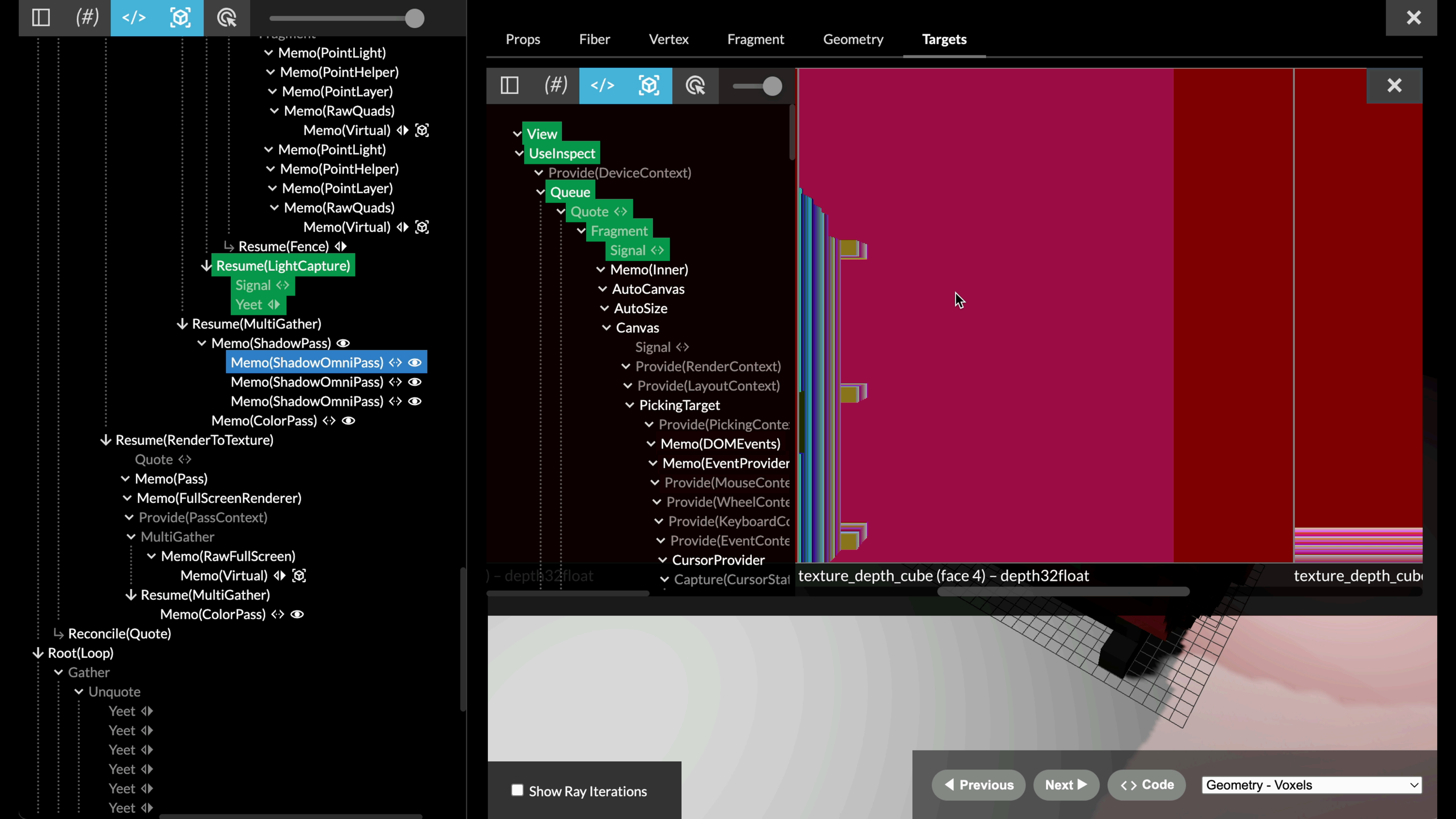
Props Fiber Vertex Fragment Geometry **Targets**

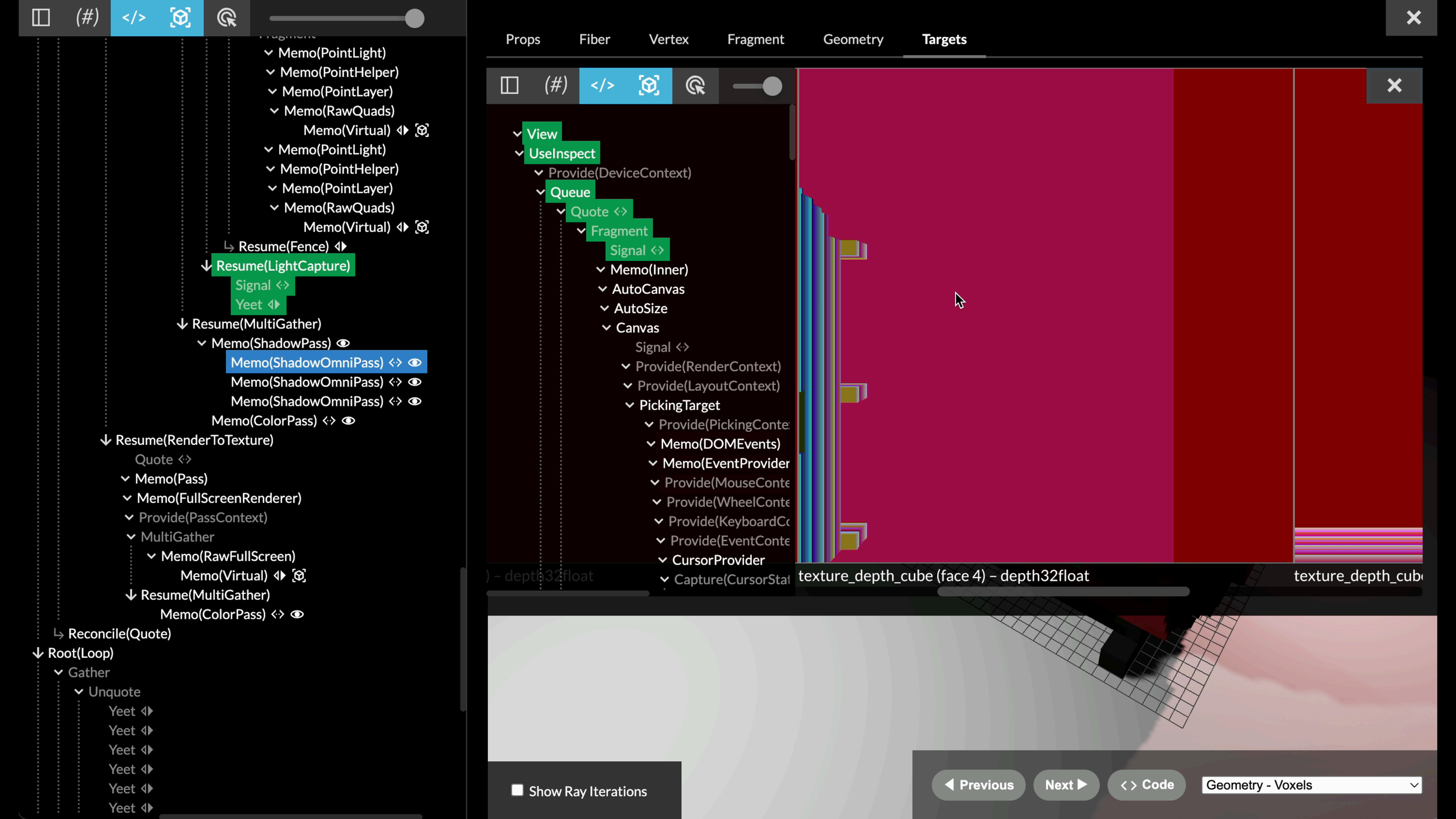
texture_depth_cube (face 5) – depth32float

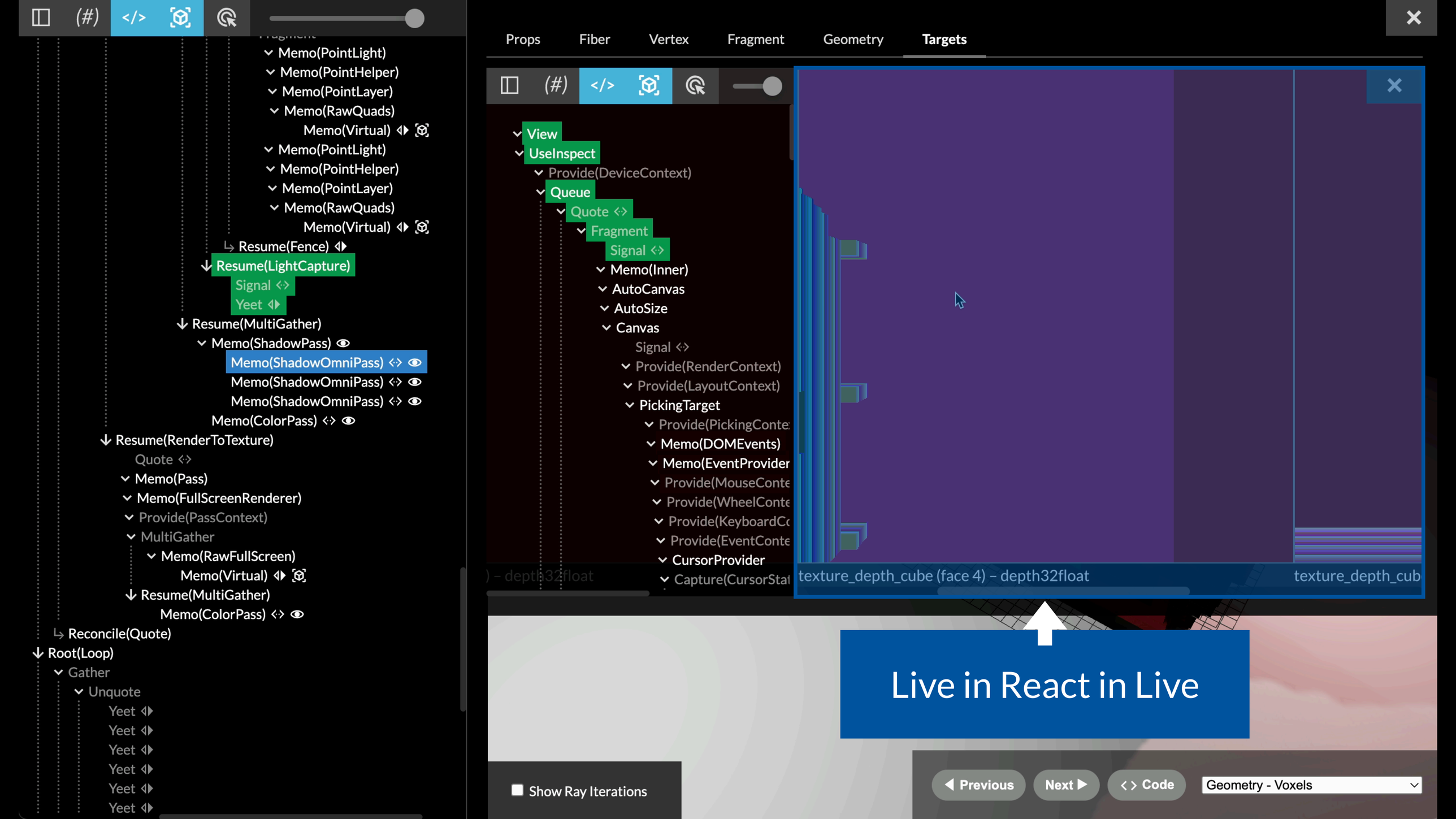
texture_depth_cube (face 6) – depth32float

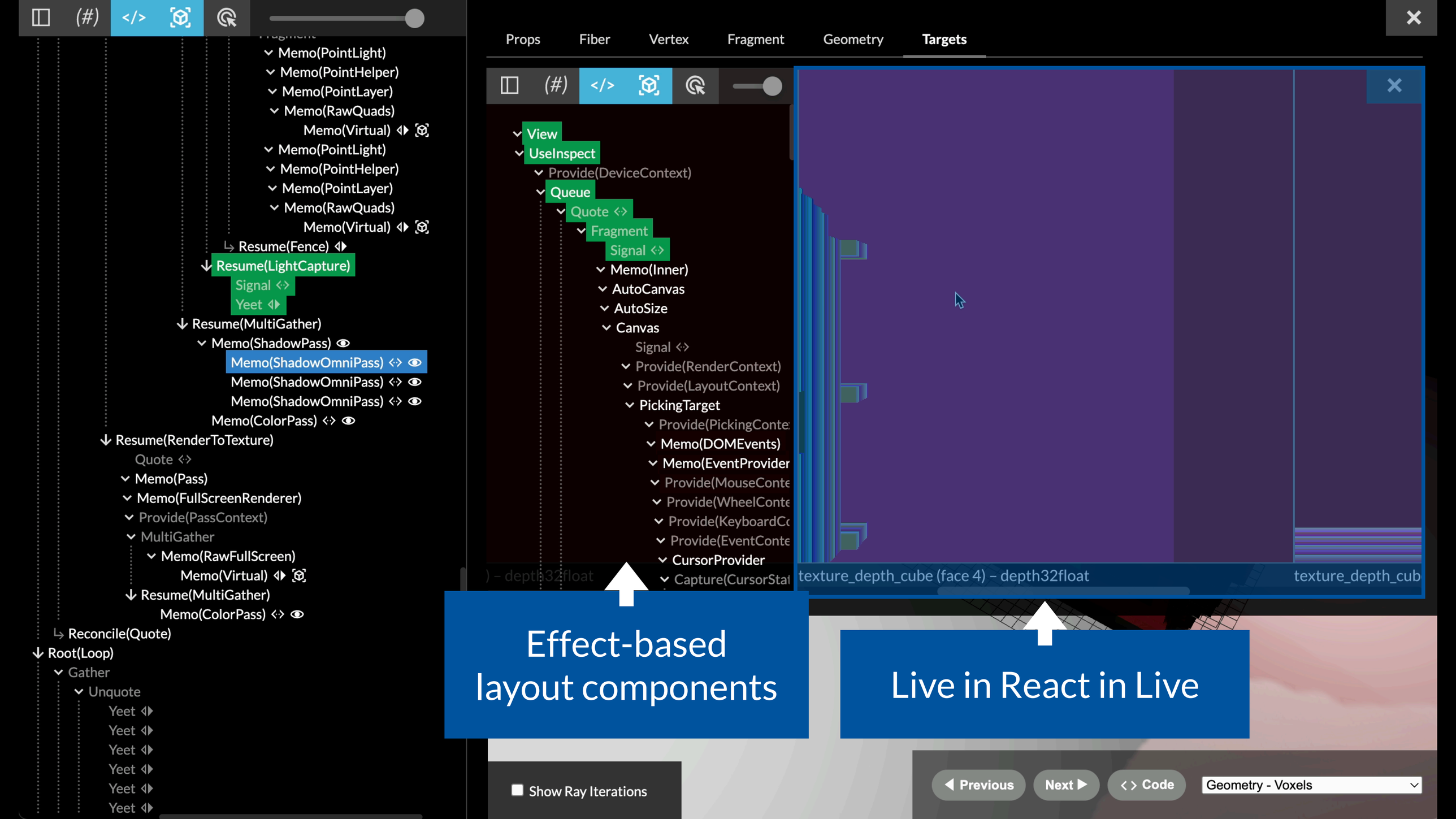
Show Ray Iterations

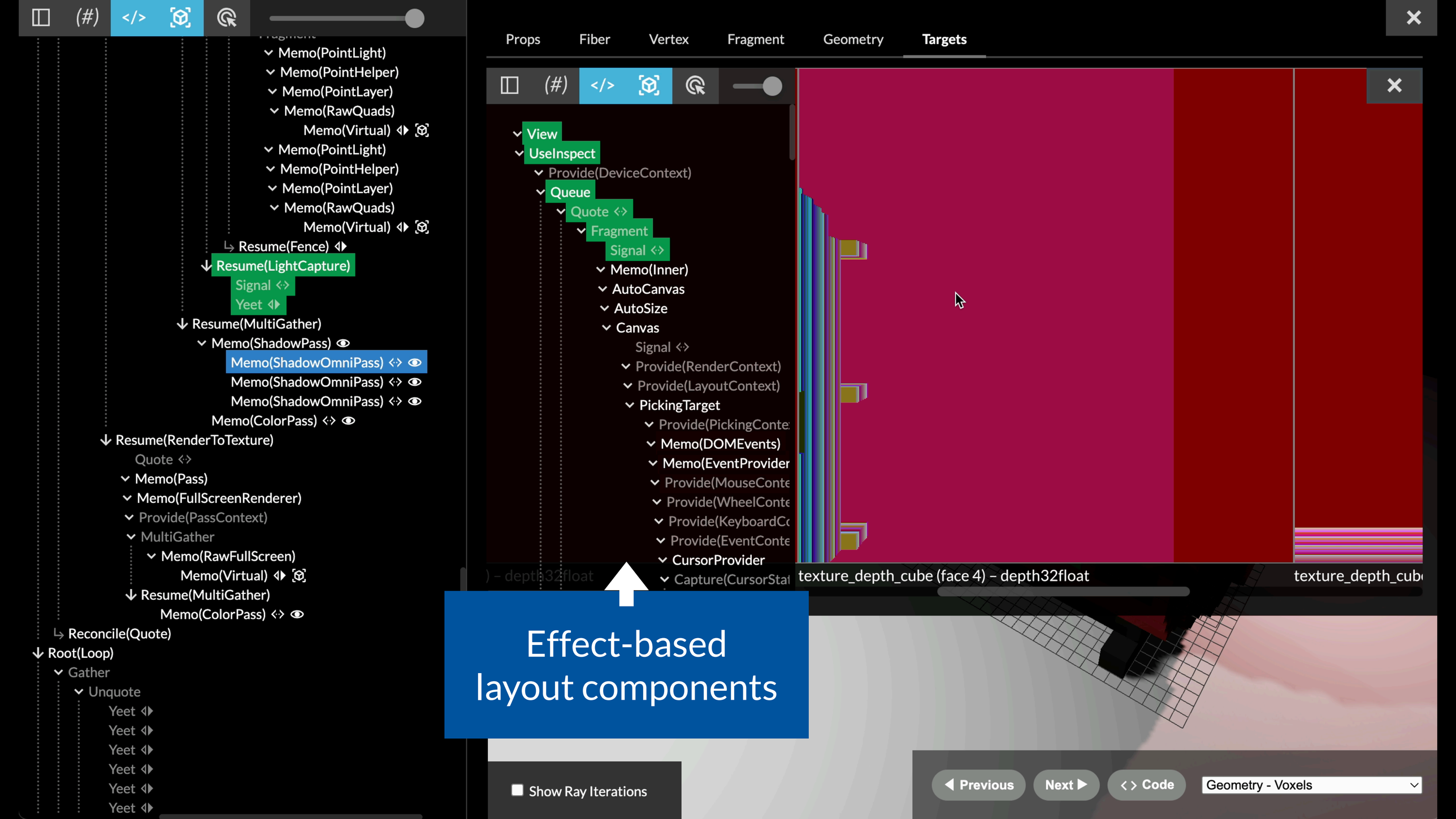
◀ Previous Next ▶ < > Code Geometry - Voxels

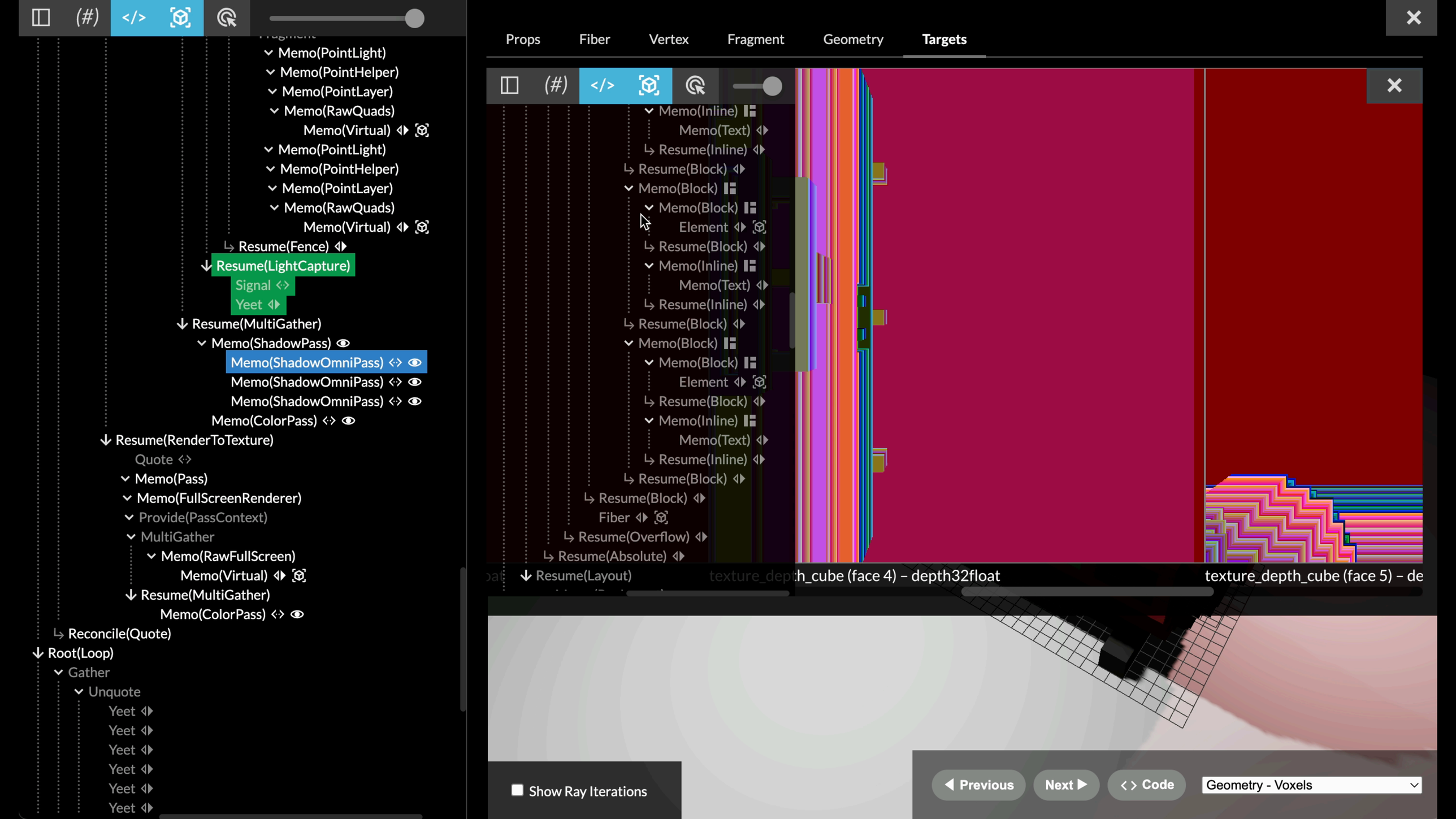

Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Props
Fiber
Vertex
Fragment
Geometry
Targets
View
UseInspect
Provide(DeviceContext)
Queue
Quote
Fragment
Signal
Memo(Inner)
AutoCanvas
AutoSize
Canvas
Signal
Provide(RenderContext)
Provide(LayoutContext)
PickingTarget
Memo(DOMEvents)
Memo(EventProvider
CursorProvider
texture_depth_cube (face 4) – depth32float
texture_depth_cub
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels

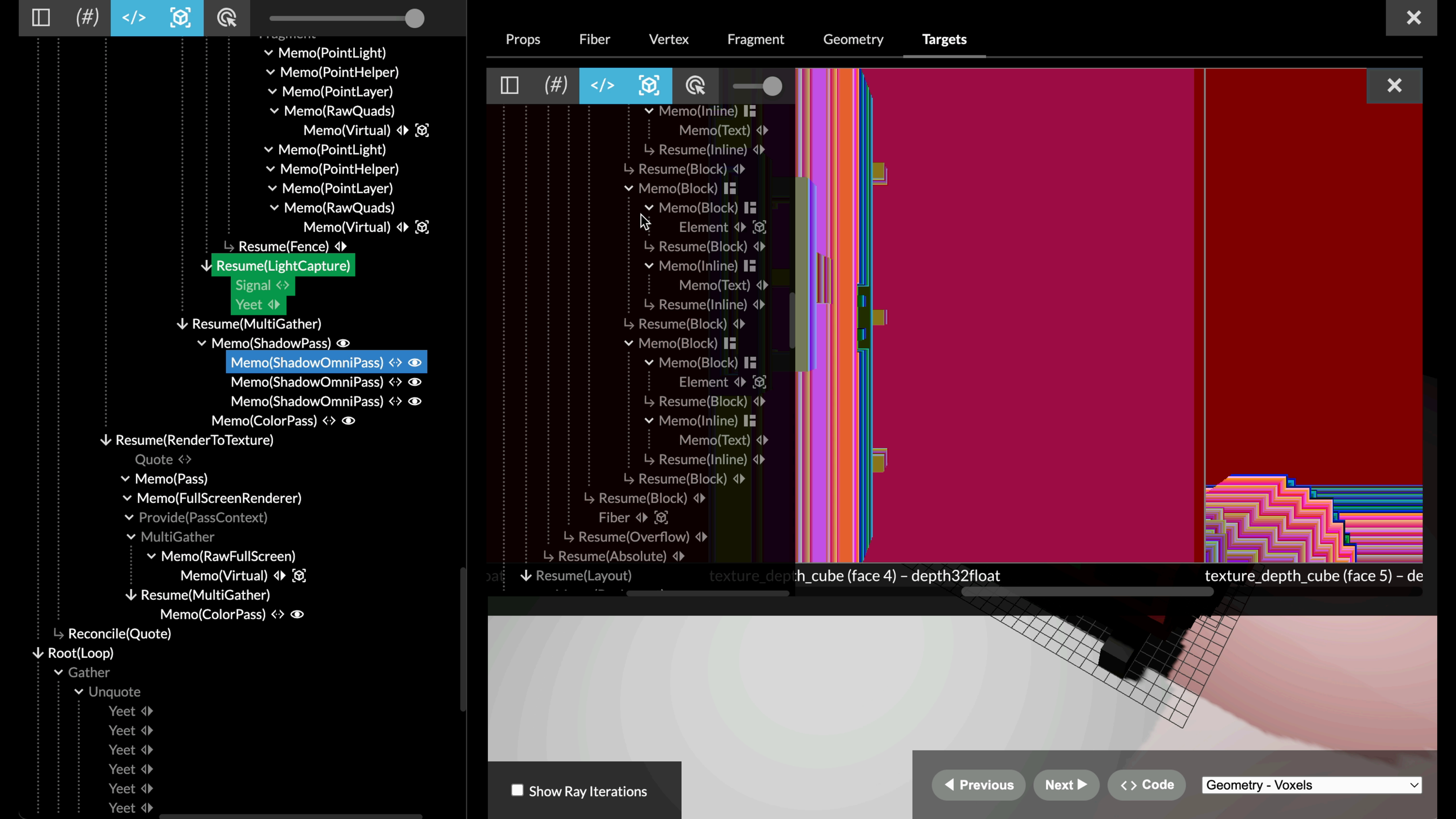

Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Props
Fiber
Vertex
Fragment
Geometry
Targets
View
UseInspect
Provide(DeviceContext)
Queue
Quote
Fragment
Signal
Memo(Inner)
AutoCanvas
AutoSize
Canvas
Signal
Provide(RenderContext)
Provide(LayoutContext)
PickingTarget
Memo(DOMEvents)
Memo(EventProvider
CursorProvider
texture_depth_cube (face 4) – depth32float
texture_depth_cub
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels

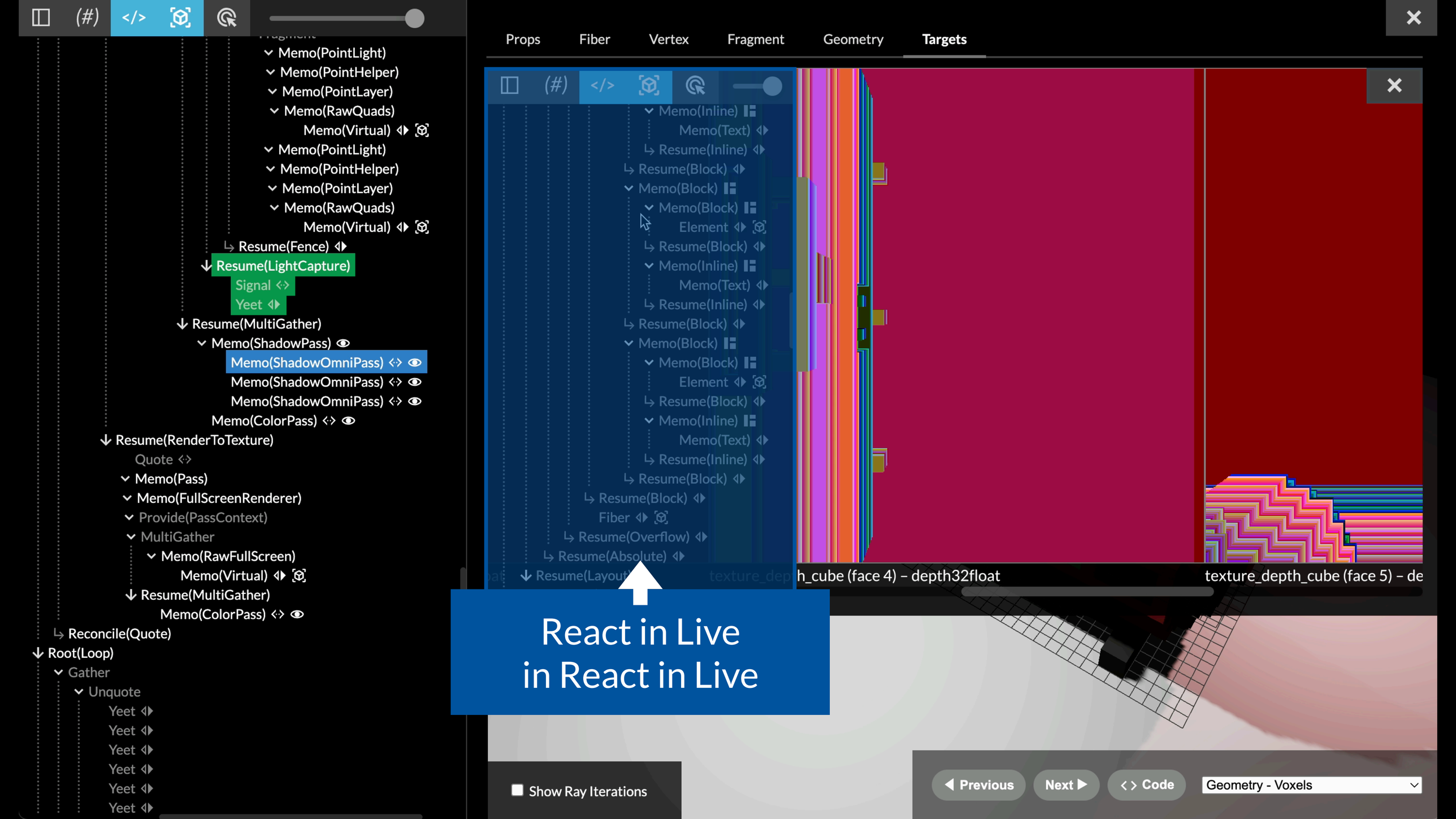

Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Props
Fiber
Vertex
Fragment
Geometry
Targets
View
UseInspect
Provide(DeviceContext)
Queue
Quote
Fragment
Signal
Memo(Inner)
AutoCanvas
AutoSize
Canvas
Signal
Provide(RenderContext)
Provide(LayoutContext)
PickingTarget
Memo(DOMEvents)
Memo(EventProvider
CursorProvider
texture_depth_cube (face 4) – depth32float
texture_depth_cub
Live in React in Live
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels

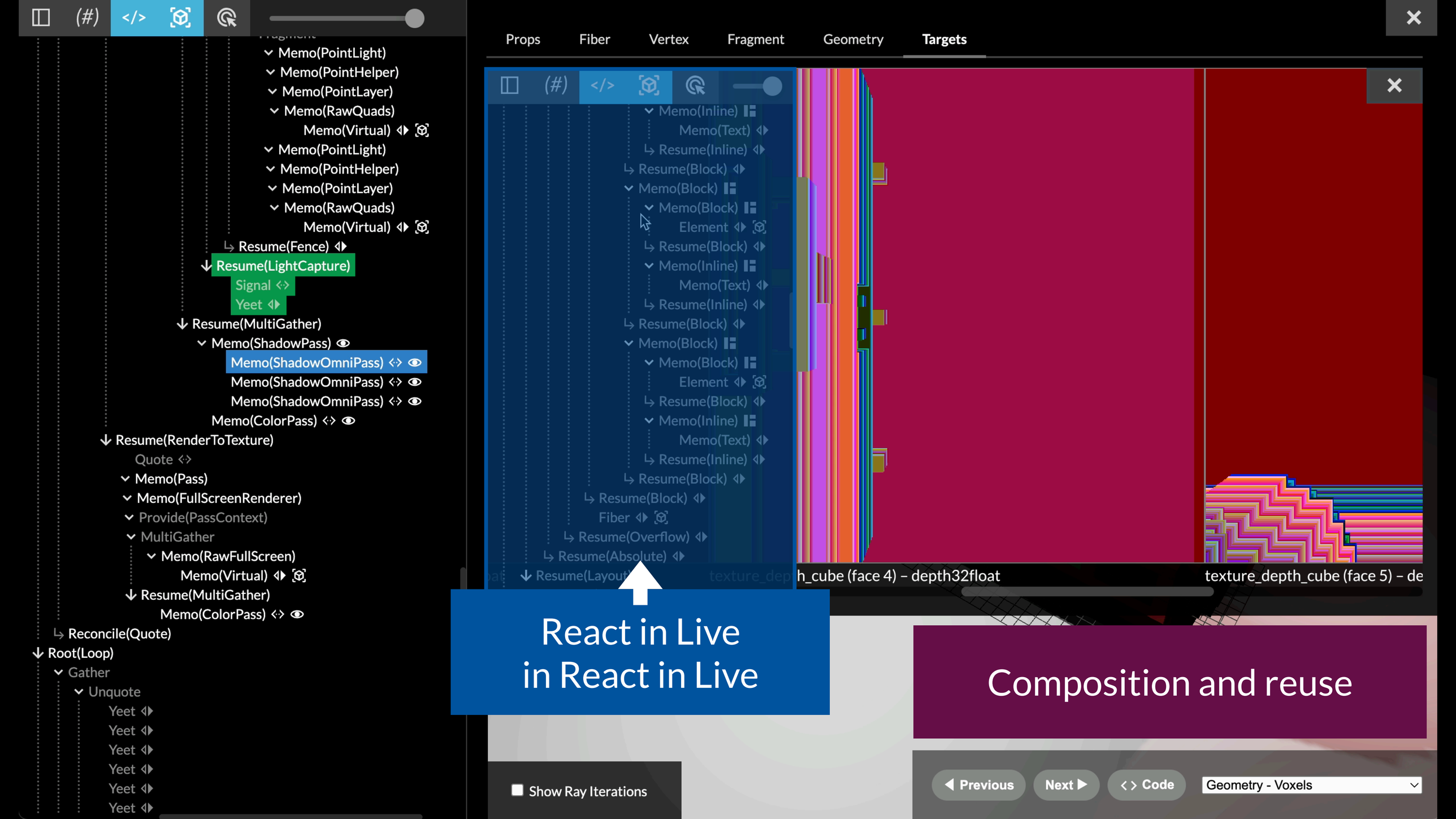

Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Resume(MultiGather)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Props
Fiber
Vertex
Fragment
Geometry
Targets
View
UseInspect
Provide(DeviceContext)
Queue
Quote
Fragment
Signal
Memo(Inner)
AutoCanvas
AutoSize
Canvas
Provide(RenderContext)
Provide(LayoutContext)
PickingTarget
Memo(DOMEvents)
CursorProvider
texture_depth_cube (face 4) – depth32float
Effect-based layout components
Live in React in Live
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels


Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Props
Fiber
Vertex
Fragment
Geometry
Targets
View
UseInspect
Provide(DeviceContext)
Queue
Quote
Fragment
Signal
Memo(Inner)
AutoCanvas
AutoSize
Canvas
Signal
Provide(RenderContext)
Provide(LayoutContext)
PickingTarget
Provide(PickingConte
Memo(DOMEvents)
Memo(EventProvider
Provide(MouseConte
Provide(WheelConte
Provide(KeyboardCo
Provide(EventConte
CursorProvider
Capture(CursorStat
texture_depth_cube (face 4) – depth32float
texture_depth_cub
Effect-based
layout components
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels


Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Props
Fiber
Vertex
Fragment
Geometry
Targets
Memo(Inline)
Memo(Text)
Resume(Inline)
Resume(Block)
Memo(Block)
Memo(Block)
Element
Resume(Block)
Memo(Inline)
Memo(Text)
Resume(Inline)
Resume(Block)
Memo(Block)
Memo(Block)
Element
Resume(Block)
Memo(Inline)
Memo(Text)
Resume(Inline)
Resume(Block)
Resume(Block)
Fiber
Resume(Overflow)
Resume(Absolute)
Resume(Layout)
texture_depth_cube (face 4) – depth32float
texture_depth_cube (face 5) – de
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels


Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Props
Fiber
Vertex
Fragment
Geometry
Targets
Memo(Inline)
Memo(Text)
Resume(Inline)
Resume(Block)
Memo(Block)
Memo(Block)
Element
Resume(Block)
Memo(Inline)
Memo(Text)
Resume(Inline)
Resume(Block)
Memo(Block)
Memo(Block)
Element
Resume(Block)
Memo(Inline)
Memo(Text)
Resume(Inline)
Resume(Block)
Resume(Block)
Fiber
Resume(Overflow)
Resume(Absolute)
Resume(Layout)
texture_depth_cube (face 4) – depth32float
texture_depth_cube (face 5) – de
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels


Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Props
Fiber
Vertex
Fragment
Geometry
Targets
Memo(Inline)
Memo(Text)
Resume(Inline)
Resume(Block)
Memo(Block)
Memo(Block)
Element
Resume(Block)
Memo(Inline)
Memo(Text)
Resume(Inline)
Resume(Block)
Memo(Block)
Memo(Block)
Element
Resume(Block)
Memo(Inline)
Memo(Text)
Resume(Inline)
Resume(Block)
Resume(Block)
Fiber
Resume(Overflow)
Resume(Absolute)
Resume(Layout
h_cube (face 4) – depth32float
texture_depth_cube (face 5) – de
React in Live
in React in Live
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels


Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Memo(PointLight)
Memo(PointHelper)
Memo(PointLayer)
Memo(RawQuads)
Memo(Virtual)
Resume(Fence)
Resume(LightCapture)
Signal
Yeet
Resume(MultiGather)
Memo(ShadowPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ShadowOmniPass)
Memo(ColorPass)
Resume(RenderToTexture)
Quote
Memo(Pass)
Memo(FullScreenRenderer)
Provide(PassContext)
MultiGather
Memo(RawFullScreen)
Memo(Virtual)
Resume(MultiGather)
Memo(ColorPass)
Reconcile(Quote)
Root(Loop)
Gather
Unquote
Yeet
Yeet
Yeet
Yeet
Yeet
Yeet
Props
Fiber
Vertex
Fragment
Geometry
Targets
Memo(Inline)
Memo(Text)
Resume(Inline)
Resume(Block)
Memo(Block)
Memo(Block)
Element
Resume(Block)
Memo(Inline)
Memo(Text)
Resume(Inline)
Resume(Block)
Memo(Block)
Memo(Block)
Element
Resume(Block)
Memo(Inline)
Memo(Text)
Resume(Inline)
Resume(Block)
Resume(Block)
Fiber
Resume(Overflow)
Resume(Absolute)
Resume(Layout
texture_dep
h_cube (face 4) – depth32float
texture_depth_cube (face 5) – de
React in Live
in React in Live
Composition and reuse
Show Ray Iterations
Previous
Next
Code
Geometry - Voxels

"How do I compose shaders?"

@use-gpu/shader

"How do I compose shaders?"

@use-gpu/shader

"How do I compose shaders?"

Shader linker with
module system in WGSL

```
use '@use-gpu/wgsl/use/types'::{ SolidVertex };
use '@use-gpu/wgsl/geometry/strip'::{ getStripIndex };
use '@use-gpu/wgsl/geometry/line'::{ getLineJoin };
use '@use-gpu/wgsl/geometry/arrow'::{ getArrowSize };
```

@use-gpu/shader

"How do I compose shaders?"

Shader linker with **module system** in WGSL

```
use '@use-gpu/wgsl/use/types'::{ SolidVertex };
use '@use-gpu/wgsl/geometry/strip'::{ getStripIndex };
use '@use-gpu/wgsl/geometry/line'::{ getLineJoin };
use '@use-gpu/wgsl/geometry/arrow'::{ getArrowSize };
```

Turn inputs into **getters**

```
array[i]    ➡    @link fn getValue(i)
```

```
@optional @link fn getWidth(i: u32) -> f32 { return 1.0; };
```

@use-gpu/shader

"How do I compose shaders?"

Shader linker with **module system** in WGSL

```
use '@use-gpu/wgsl/use/types'::{ SolidVertex };
use '@use-gpu/wgsl/geometry/strip'::{ getStripIndex };
use '@use-gpu/wgsl/geometry/line'::{ getLineJoin };
use '@use-gpu/wgsl/geometry/arrow'::{ getArrowSize };
```

Turn inputs into **getters**

```
array[i]  ➡  @link fn getValue(i)
@optional @link fn getWidth(i: u32) -> f32 { return 1.0; };
```

Shader **imports** in TypeScript

```
import { getLineSegment } from '@use-gpu/wgsl/geometry/segment.wgsl';
import { getLineVertex } from '@use-gpu/wgsl/instance/vertex/line.wgsl';
```

@use-gpu/shader

"How do I compose shaders?"

Shader linker with **module system** in WGSL

```
use '@use-gpu/wgsl/use/types'::{ SolidVertex };
use '@use-gpu/wgsl/geometry/strip'::{ getStripIndex };
use '@use-gpu/wgsl/geometry/line'::{ getLineJoin };
use '@use-gpu/wgsl/geometry/arrow'::{ getArrowSize };
```

Turn inputs into **getters**

```
array[i]  ➡  @link fn getValue(i)
@optional @link fn getWidth(i: u32) -> f32 { return 1.0; };
```

Shader **imports** in TypeScript

```
import { getLineSegment } from '@use-gpu/wgsl/geometry/segment.wgsl';
import { getLineVertex } from '@use-gpu/wgsl/instance/vertex/line.wgsl';
```

Bind args by **handle**, **reference** or **lambda**

```
const getVertex = useShader(getLineVertex, [
  positions, scissor,
  uvs, sts,
  segments, colors, widths, depths, zBiases,
]);
```

@use-gpu/shader

"How do I compose shaders?"

Shader linker with **module system** in WGSL

```
use '@use-gpu/wgsl/use/types'::{ SolidVertex };
use '@use-gpu/wgsl/geometry/strip'::{ getStripIndex };
use '@use-gpu/wgsl/geometry/line'::{ getLineJoin };
use '@use-gpu/wgsl/geometry/arrow'::{ getArrowSize };
```

Turn inputs into **getters**

```
array[i]  ➡  @link fn getValue(i)
@optional @link fn getWidth(i: u32) -> f32 { return 1.0; };
```

Shader **imports** in TypeScript

```
import { getLineSegment } from '@use-gpu/wgsl/geometry/segment.wgsl';
import { getLineVertex } from '@use-gpu/wgsl/instance/vertex/line.wgsl';
```

Bind args by **handle**, **reference** or **lambda**

```
const getVertex = useShader(getLineVertex, [
  positions, scissor,
  uvs, sts,
  segments, colors, widths, depths, zBiases,
]);
```

Closures

@use-gpu/shader

"How do I compose shaders?"

```
181
182
183  //// @link use/shadow.wgsl _05_
184
185  @group(1) @binding(1) var _05_lightTexture: texture_depth_2d_array;
186  @group(1) @binding(2) var _05_lightSampler: sampler_comparison;
187
188  fn _05_sampleShadow(uv: vec2<f32>, index: u32, level: f32) -> f32 {
189    return textureSampleCompareLevel(_05_lightTexture, _05_lightSampler, uv, index, level);
190  }
191
192  //// @link material/pbr-apply.wgsl _06_
193
194  fn _06_applyPBRMaterial(
195    N: vec3<f32>,
196    L: vec3<f32>,
197    V: vec3<f32>,
198    surface: _02_SurfaceFragment,
199  ) -> vec3<f32> {
200    return _03_PBR(N, L, V, surface.albedo.xyz, surface.material.x, surface.material.y);
201  }
202
203  //// @link @access [getMatrix getRayMatrix getNormalMatrix getSize getIsInside getInsideOrigin getPBR getPalette
204
205  fn _VT_4_getMatrix() -> mat4x4<f32> {
206    return _VT_Uniform._VT_4_getMatrix;
207  }
208
209  fn _VT_4_getRayMatrix() -> mat3x3<f32> {
```

@use-gpu/shader

"How do I compose shaders?"

```
181
182
183  //// @link use/shadow.wgsl _05_
184
185  @group(1) @binding(1) var _05_lightTexture: texture_depth_2d_array;
186  @group(1) @binding(2) var _05_lightSampler: sampler_comparison;
187
188  fn _05_sampleShadow(uv: vec2<f32>, index: u32, level: f32) -> f32 {
189    return textureSampleCompareLevel(_05_lightTexture, _05_lightSampler, uv, index, level);
190  }
191
192  //// @link material/pbr-apply.wgsl _06_
193
194  fn _06_applyPBRMaterial(
195    N: vec3<f32>,
196    L: vec3<f32>,
197    V: vec3<f32>,
198    surface: _02_SurfaceFragment,
199  ) -> vec3<f32> {
200    return _03_PBR(N, L, V, surface.albedo.xyz, surface.material.x, surface.material.y);
201  }
202
203  //// @link @access [getMatrix getRayMatrix getNormalMatrix getSize getIsInside getInsideOrigin getPBR getPalette
204
205  fn _VT_4_getMatrix() -> mat4x4<f32> {
206    return _VT_Uniform._VT_4_getMatrix;
207  }
208
209  fn _VT_4_getRayMatrix() -> mat3x3<f32> {
```

Links modules

@use-gpu/shader

"How do I compose shaders?"

```
181
182
183 //// @link use/shadow.wgsl _05_
184
185 @group(1) @binding(1) var _05_lightTexture: texture_depth_2d_array;
186 @group(1) @binding(2) var _05_lightSampler: sampler_comparison;
187
188 fn _05_sampleShadow(uv: vec2<f32>, index: u32, level: f32) -> f32 {
189   return textureSampleCompareLevel(_05_lightTexture, _05_lightSampler, uv, index, level);
190 }
191
192 //// @link material/pbr-apply.wgsl _06_
193
194 fn _06_applyPBRMaterial(
195   N: vec3<f32>,
196   L: vec3<f32>,
197   V: vec3<f32>,
198   surface: _02_SurfaceFragment,
199 ) -> vec3<f32> {
200   return _03_PBR(N, L, V, surface.albedo.xyz, surface.material.x, surface.material.y);
201 }
202
203 //// @link @access [getMatrix getRayMatrix getNormalMatrix getSize getIsInside getInsideOrigin getPBR getPalette
204
205 fn _VT_4_getMatrix() -> mat4x4<f32> {
206   return _VT_Uniform._VT_4_getMatrix;
207 }
208
209 fn _VT_4_getRayMatrix() -> mat3x3<f32> {
```

Links modules

Aggregates structs

@use-gpu/shader

"How do I compose shaders?"

```
181
182
183 //// @link use/shadow.wgsl _05_
184
185 @group(1) @binding(1) var _05_lightTexture: texture_depth_2d_array;
186 @group(1) @binding(2) var _05_lightSampler: sampler_comparison;
187
188 fn _05_sampleShadow(uv: vec2<f32>, index: u32, level: f32) -> f32 {
189   return textureSampleCompareLevel(_05_lightTexture, _05_lightSampler, uv, index, level);
190 }
191
192 //// @link material/pbr-apply.wgsl _06_
193
194 fn _06_applyPBRMaterial(
195   N: vec3<f32>,
196   L: vec3<f32>,
197   V: vec3<f32>,
198   surface: _02_SurfaceFragment,
199 ) -> vec3<f32> {
200   return _03_PBR(N, L, V, surface.albedo.xyz, surface.material.x, surface.material.y);
201 }
202
203 //// @link @access [getMatrix getRayMatrix getNormalMatrix getSize getIsInside getInsideOrigin getPBR getPalette
204
205 fn _VT_4_getMatrix() -> mat4x4<f32> {
206   return _VT_Uniform._VT_4_getMatrix;
207 }
208
209 fn _VT_4_getRayMatrix() -> mat3x3<f32> {
```

Links modules

Aggregates structs

Generates boilerplate and bindings

@use-gpu/shader

"How do I compose shaders?"

```
6
7   //// @link @virtual _01_
8
9   struct _VT_Type {
10    _VT_1_getSegmentCount: f32,
11    _VT_2_getTransformMatrix: mat4x4<f32>,
12    _VT_3_getFlip: vec2<f32>,
13    _VT_3_getOffset: vec2<f32>,
14  };
15  @group(1) @binding(4) var<uniform> _VT_Uniform: _VT_Type;
16
17  //// @link @struct getColorsZBiasesWidthsSizesDepths _02_
18
19  struct _02_getColorsZBiasesWidthsSizesDepths {
20    colors: vec4<f32>,
21    zBiases: f32,
22    widths: f32,
23    sizes: f32,
24    depths: f32,
25  };
26
27  //// @link @access [getTransformMatrix] [wvkim01pej] _VT_2_
28
29  fn _VT_2_getTransformMatrix(a: u32) -> mat4x4<f32> {
30    return _VT_Uniform._VT_2_getTransformMatrix;
31  }
32
33  //// @link @access [getFlip getOffset] [lfqwh2k27v] _VT_3_
34
```

@use-gpu/shader

"How do I compose shaders?"

```
6
7   //// @link @virtual _01_
8
9   struct _VT_Type {
10    _VT_1_getSegmentCount: f32,
11    _VT_2_getTransformMatrix: mat4x4<f32>,
12    _VT_3_getFlip: vec2<f32>,
13    _VT_3_getOffset: vec2<f32>,
14  };
15  @group(1) @binding(4) var<uniform> _VT_Uniform: _VT_Type;
16
17  //// @link @struct getColorsZBiasesWidthsSizesDepths _02_
18
19  struct _02_getColorsZBiasesWidthsSizesDepths {
20    colors: vec4<f32>,
21    zBiases: f32,
22    widths: f32,
23    sizes: f32,
24    depths: f32,
25  };
26
27  //// @link @access [getTransformMatrix] [wvkim01pej] _VT_2_
28
29  fn _VT_2_getTransformMatrix(a: u32) -> mat4x4<f32> {
30    return _VT_Uniform._VT_2_getTransformMatrix;
31  }
32
33  //// @link @access [getFlip getOffset] [lfqwh2k27v] _VT_3_
34
```

So much boilerplate

@use-gpu/shader

"How do I compose shaders?"

```
  return _VT_Uniform._VT_3_getFlip;
}

fn _VT_3_getOffset() → vec2<f32> {
  return _VT_Uniform._VT_3_getOffset;
}

//// @link @struct getPositionsAnchorsTrims _05_

struct _05_getPositionsAnchorsTrims {
  positions: vec4<f32>,
  anchors: vec4<u32>,
  trims: vec4<u32>,
};

//// @link @access [getColorsZBiasesWidthsSizesDepths] [fi2lsvtdfc] _06_

@group(1) @binding(3) var<storage> _06_getColorsZBiasesWidthsSizesDepths: array

//// @link @access [instances] [3n6yiygoq4] _07_

@group(1) @binding(2) var<storage> _07_instancesStorage: array<u32>;

fn _07_instances(a: u32) → u32 {
  return _07_instancesStorage[a];
}

//// @link transform/cartesian.wgsl _08_

fn _08_getCartesianPosition(vector: vec4<f32>) → vec4<f32> {
  return _VT_2_getTransformMatrix(0) * vec4<f32>(vector.xyz, 1.0);
}
```

So much boilerplate

@use-gpu/shader

"How do I compose shaders?"

```
  return _VT_Uniform._VT_3_getFlip;
}

fn _VT_3_getOffset() → vec2<f32> {
  return _VT_Uniform._VT_3_getOffset;
}

//// @link @struct getPositionsAnchorsTrims _05_

struct _05_getPositionsAnchorsTrims {
  positions: vec4<f32>,
  anchors: vec4<u32>,
  trims: vec4<u32>,
};

//// @link @access [getColorsZBiasesWidthsSizesDepths] [fi2lsvtdfc] _06_

@group(1) @binding(3) var<storage> _06_getColorsZBiasesWidthsSizesDepths: array

//// @link @access [instances] [3n6yiygoq4] _07_

@group(1) @binding(2) var<storage> _07_instancesStorage: array<u32>;

fn _07_instances(a: u32) → u32 {
  return _07_instancesStorage[a];
}

//// @link transform/cartesian.wgsl _08_

fn _08_getCartesianPosition(vector: vec4<f32>) → vec4<f32> {
  return _VT_2_getTransformMatrix(0) * vec4<f32>(vector.xyz, 1.0);
}
```

So much boilerplate

All at run-time

@use-gpu/shader

"How do I compose shaders?"

Raytrace a .vox model using the Voxel package. Composes with the existing lighting and shadow components.

Props | Fiber | Vertex | **Fragment** | Geometry

Constants

- **_VT_1_getTransformMatrix** {uniform: {…}, constant: {…}}
- **_VT_2_getNormalMatrix** {uniform: {…}, constant: {…}}
- **_VT_3_getSize** {uniform: {…}, constant: (…) shape[0].size, resolved: [18, 50, 22]}
- **_VT_4_getMatrix** {uniform: {…}, constant: [-1, 1.2246e-16, 0, 0, 0, 3.4229e-8, 1, 0, 1.2246e-16, …
- **_VT_4_getRayMatrix** {uniform: {…}, constant: [-1.0000001192092896, -4.1918e-24, 1.2246e-16, 1.2246e-…
- **_VT_4_getNormalMatrix** {uniform: {…}, constant: [-1.0000001192092896, 1.2246e-16, -4.1918e-24, -4.1918e…
- **_VT_4_getSize** {uniform: {…}, constant: (…) shape[0].size, resolved: [18, 50, 22]}
- **_VT_4_getIsInside** {uniform: {…}, constant: {…}}
- **_VT_4_getInsideOrigin** {uniform: {…}, constant: {…}}

Bindings

- **getPositionsNormalsUvs** {uniform: {…}, storage: {…}}
- **getPBR** {uniform: {…}, storage: {…}}
- **getPalette** {uniform: {…}, texture: {…}}
- **getTexture0** {uniform: {…}, texture: {…}}
- **getTexture1** {uniform: {…}, texture: {…}}
- **getTexture2** {uniform: {…}, texture: {…}}

Shader (597dplhrsf-wj4qku02oe-1)

Hot Reload ⌘ + S

```
1  const SHADOW_PAGE = 4096;
2  const DEBUG_STEPS = false;
3  const MIP_LEVELS = 3;
4  const SDF_LEVEL = -1;
```

@use-gpu/shader

"How do I compose shaders?"

Raytrace a .vox model using the Voxel package. Composes with the existing lighting and shadow components.

Props | Fiber | Vertex | **Fragment** | Geometry

Constants

- **_VT_1_getTransformMatrix** {uniform: {…}, constant: {…}}
- **_VT_2_getNormalMatrix** {uniform: {…}, constant: {…}}
- **_VT_3_getSize** {uniform: {…}, constant: (…) shape[0].size, resolved: [18, 50, 22]}
- **_VT_4_getMatrix** {uniform: {…}, constant: [-1, 1.2246e-16, 0, 0, 0, 3.4229e-8, 1, 0, 1.2246e-16, …
- **_VT_4_getRayMatrix** {uniform: {…}, constant: [-1.0000001192092896, -4.1918e-24, 1.2246e-16, 1.2246e-…
- **_VT_4_getNormalMatrix** {uniform: {…}, constant: [-1.0000001192092896, 1.2246e-16, -4.1918e-24, -4.1918e…
- **_VT_4_getSize** {uniform: {…}, constant: (…) shape[0].size, resolved: [18, 50, 22]}
- **_VT_4_getIsInside** {uniform: {…}, constant: {…}}
- **_VT_4_getInsideOrigin** {uniform: {…}, constant: {…}}

Bindings

- **getPositionsNormalsUvs** {uniform: {…}, storage: {…}}
- **getPBR** {uniform: {…}, storage: {…}}
- **getPalette** {uniform: {…}, texture: {…}}
- **getTexture0** {uniform: {…}, texture: {…}}
- **getTexture1** {uniform: {…}, texture: {…}}
- **getTexture2** {uniform: {…}, texture: {…}}

Shader (597dplhrsf-wj4qku02oe-1)

Hot Reload ⌘ + S

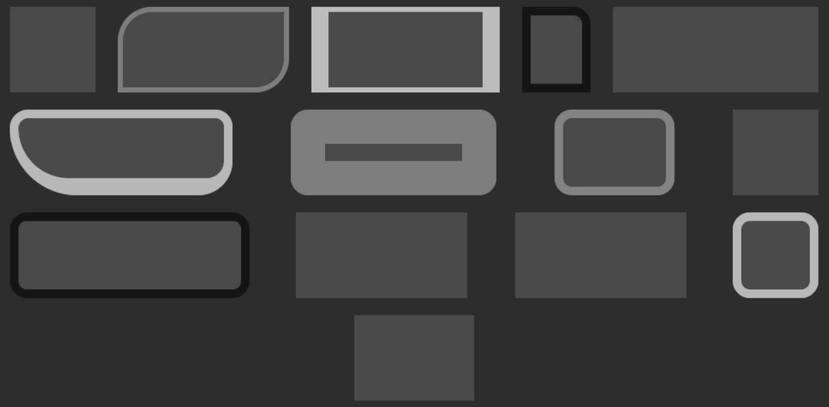
```
1  const SHADOW_PAGE = 4096;
2  const DEBUG_STEPS = false;
3  const MIP_LEVELS = 3;
4  const SDF_LEVEL = -1;
```

Inspect generated shader + bindings

Use.GPU

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.



- (•) Interact
- ( ) Pan & Zoom
- ( ) SDFs

Lorem ipsum ⭐✨🍕

Dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Fugiat nulla pariatur

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud Aute Cupiditat Aliquip exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem x ipsum dolor sit amet, x consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Sit amet nostrum labore

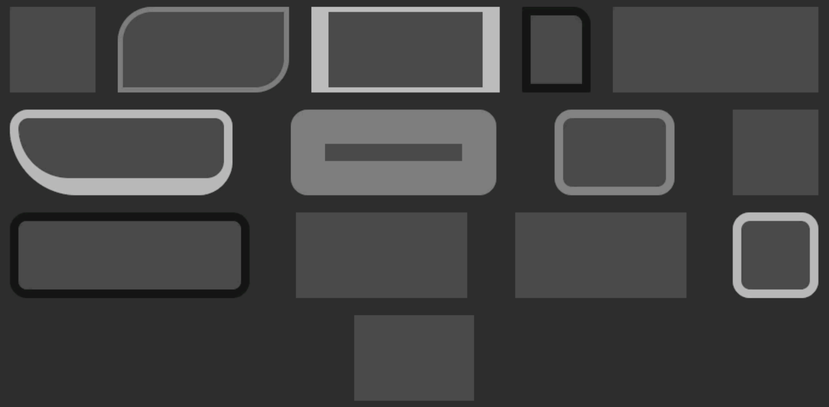
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore

◀ Previous | Next ▶ | < > Code | Layout - Box model

Use.GPU

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.



- (•) Interact
- ( ) Pan & Zoom
- ( ) SDFs

Lorem ipsum ⭐✨🍕

Dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Fugiat nulla pariatur

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud Aute Cupiditat Aliquip exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem x ipsum dolor sit amet, x consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Sit amet nostrum labore

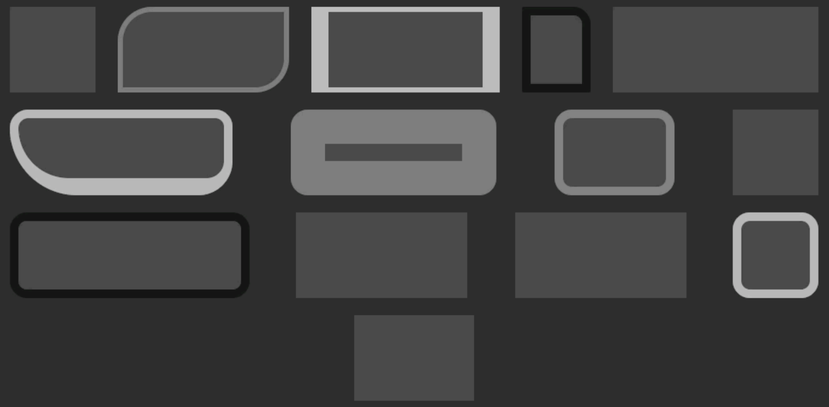
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore

◀ Previous | Next ▶ | < > Code | Layout - Box model

Use.GPU

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.



- Interact
- Pan & Zoom
- SDFs

Lorem ipsum ⭐✨🍕

Dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Fugiat nulla pariatur

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud Aute Cupiditat Aliquip exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui offi

Lorem x ipsum dolor sit amet, x consectetur adipisicing eli
dolore magna aliqua.

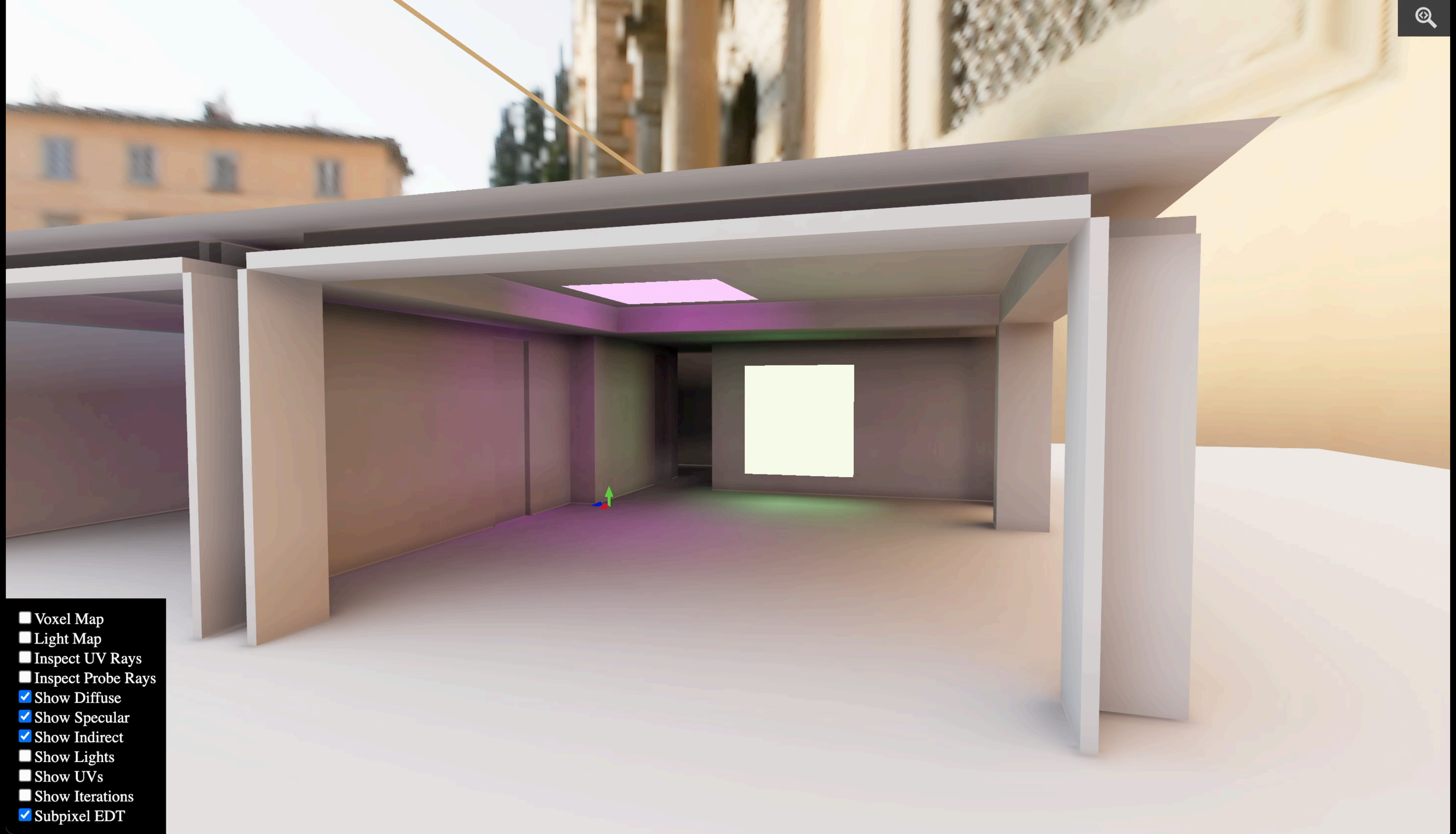
Ut enim ad minim veniam, quis nostrud exercitation ullam
Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur
sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

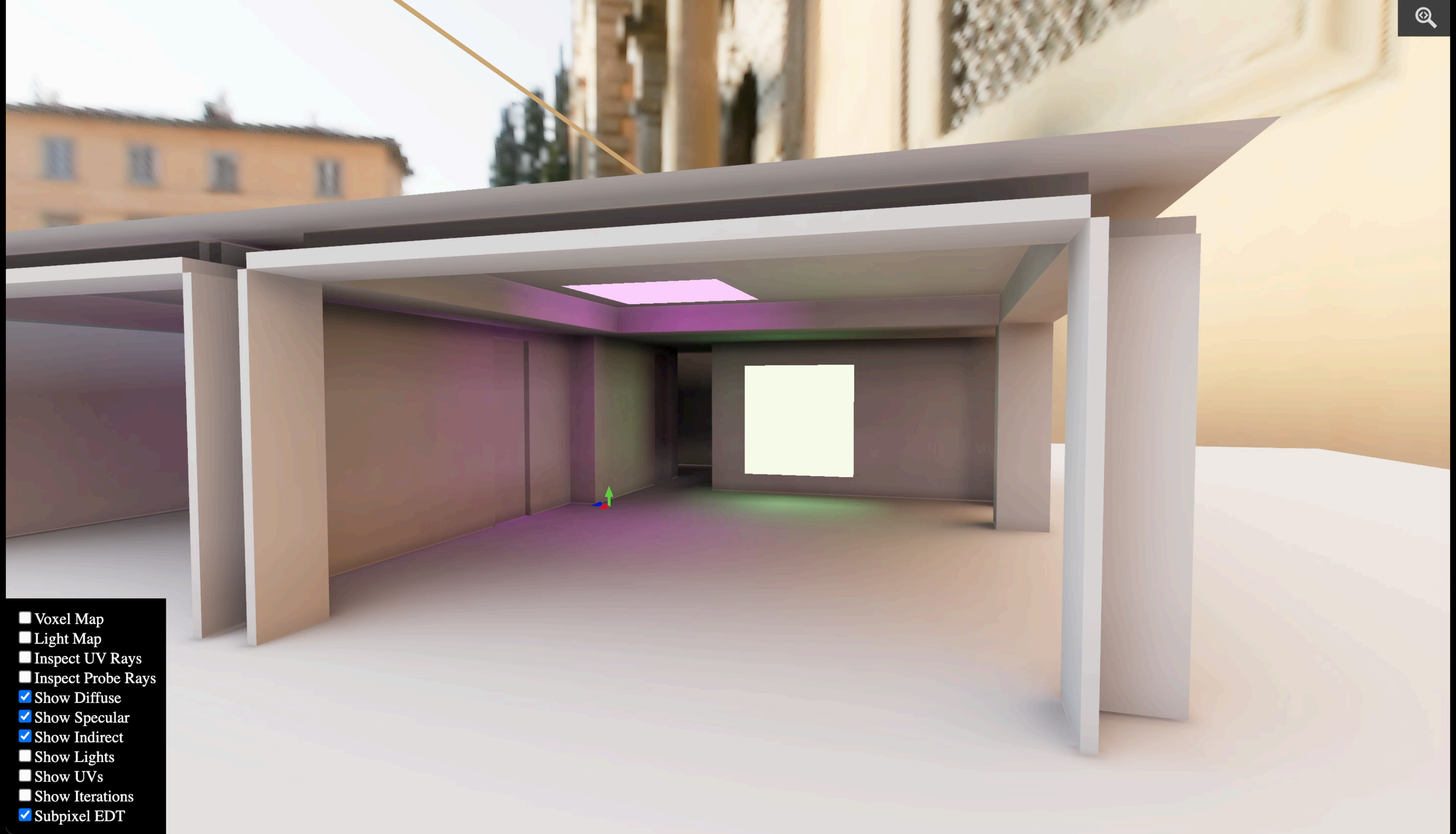
Rendering with distance fields

Sit amet nostrum labore

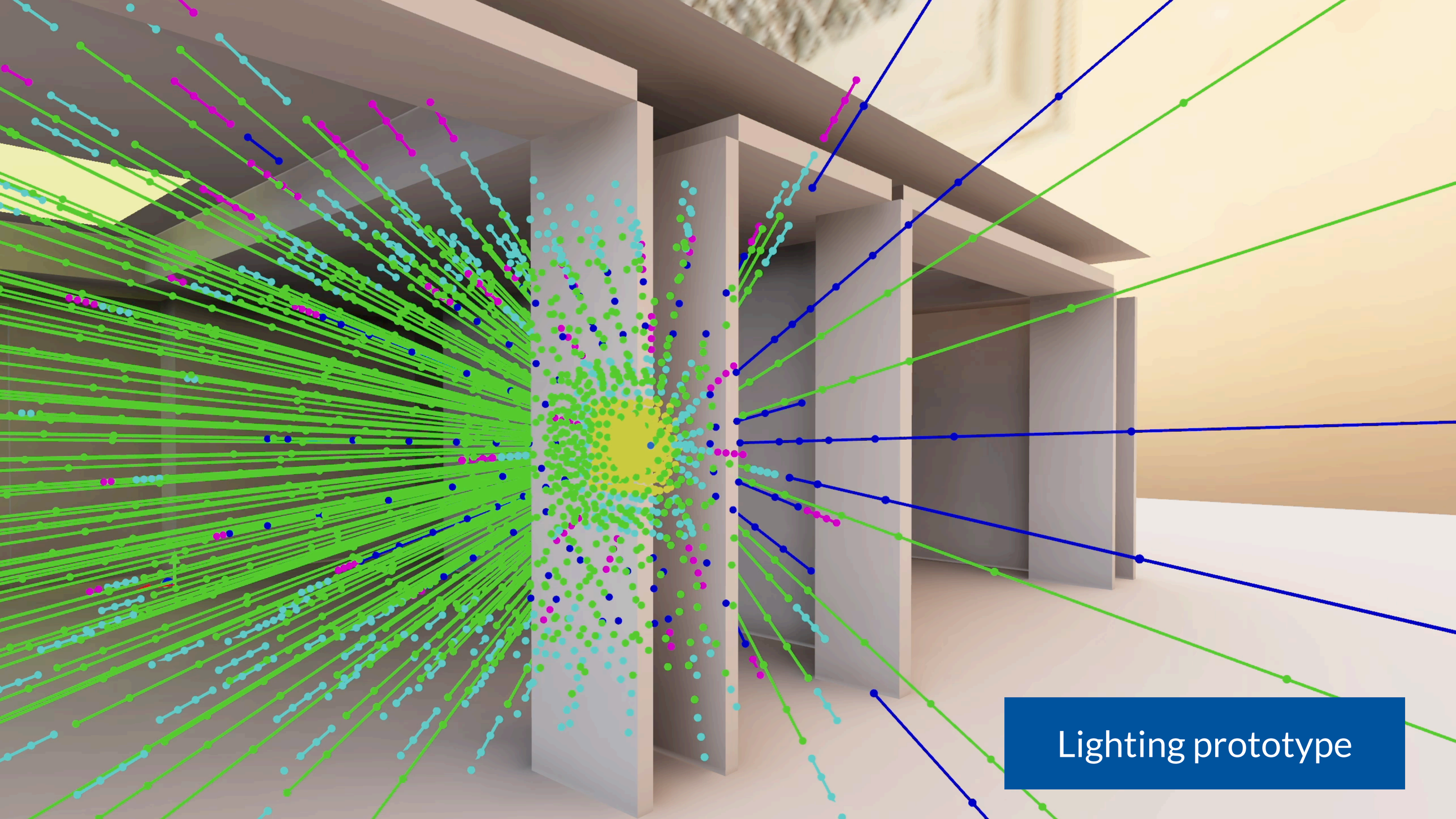
◀ Previous | Next ▶ | <> Code | Layout - Box model

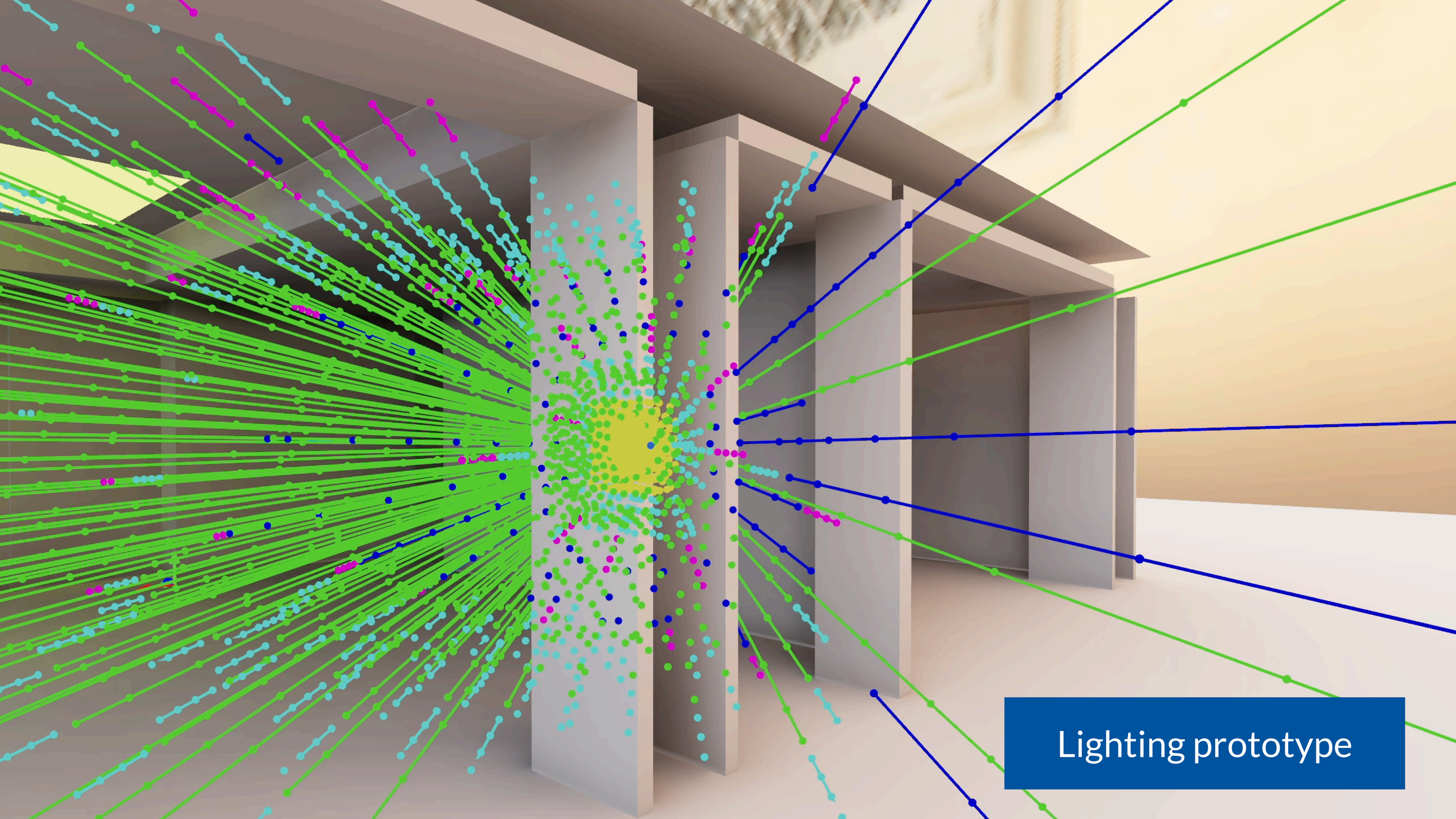
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore

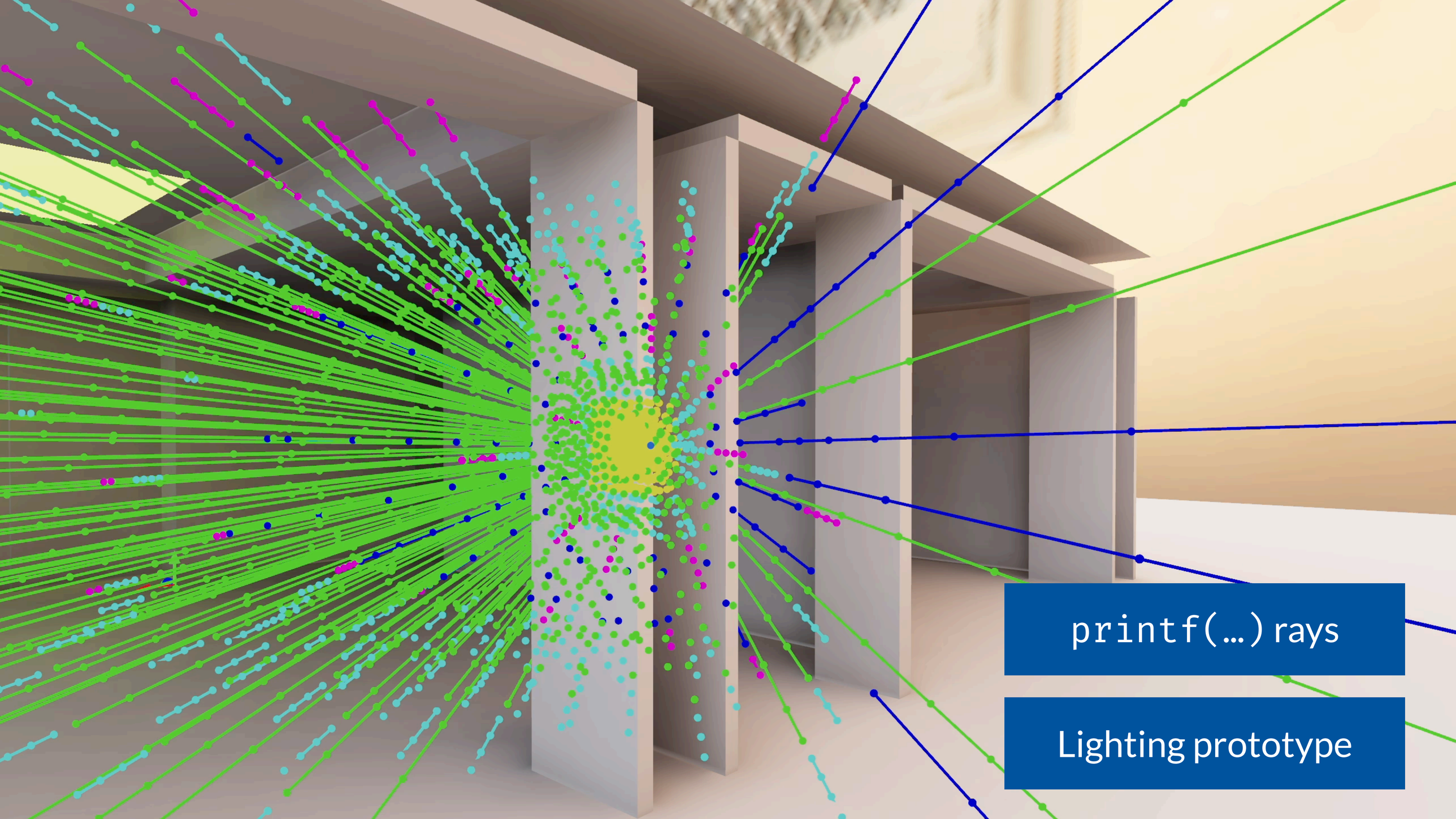

Voxel Map
Light Map
Inspect UV Rays
Inspect Probe Rays
Show Diffuse
Show Specular
Show Indirect
Show Lights
Show UVs
Show Iterations
Subpixel EDT

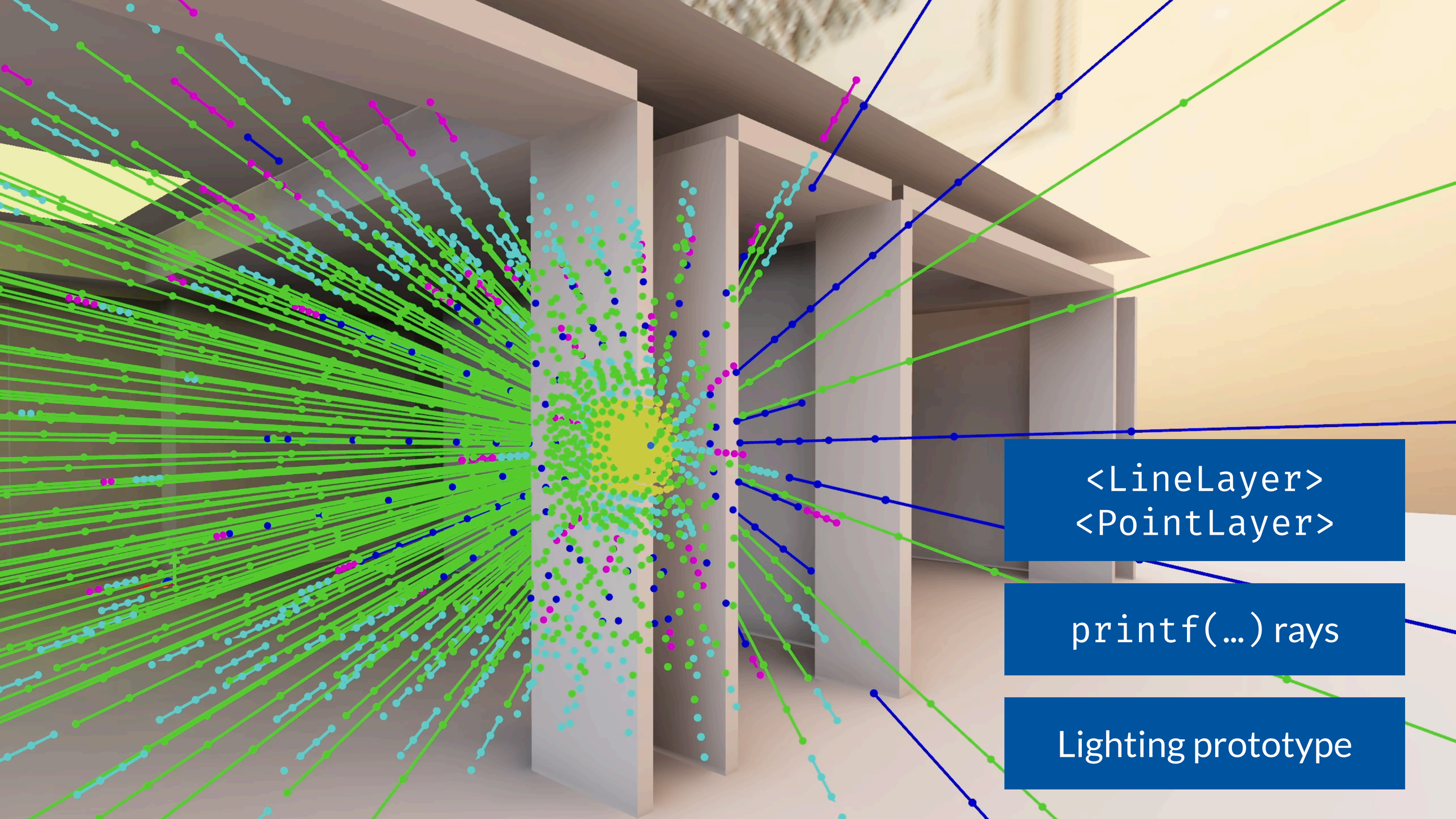

Voxel Map
Light Map
Inspect UV Rays
Inspect Probe Rays
Show Diffuse
Show Specular
Show Indirect
Show Lights
Show UVs
Show Iterations
Subpixel EDT

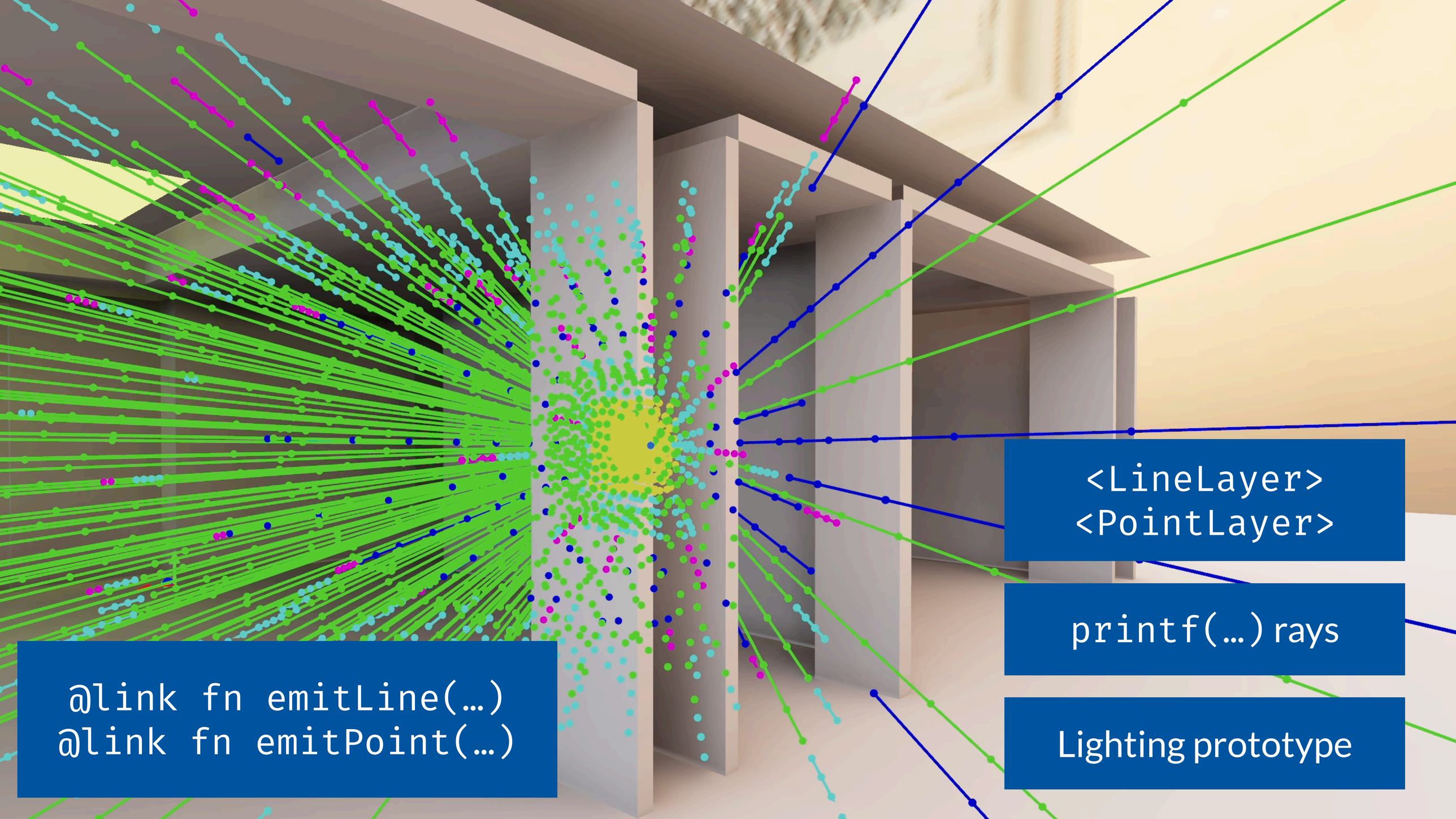

Voxel Map
Light Map
Inspect UV Rays
Inspect Probe Rays
Show Diffuse
Show Specular
Show Indirect
Show Lights
Show UVs
Show Iterations
Subpixel EDT
Lighting prototype

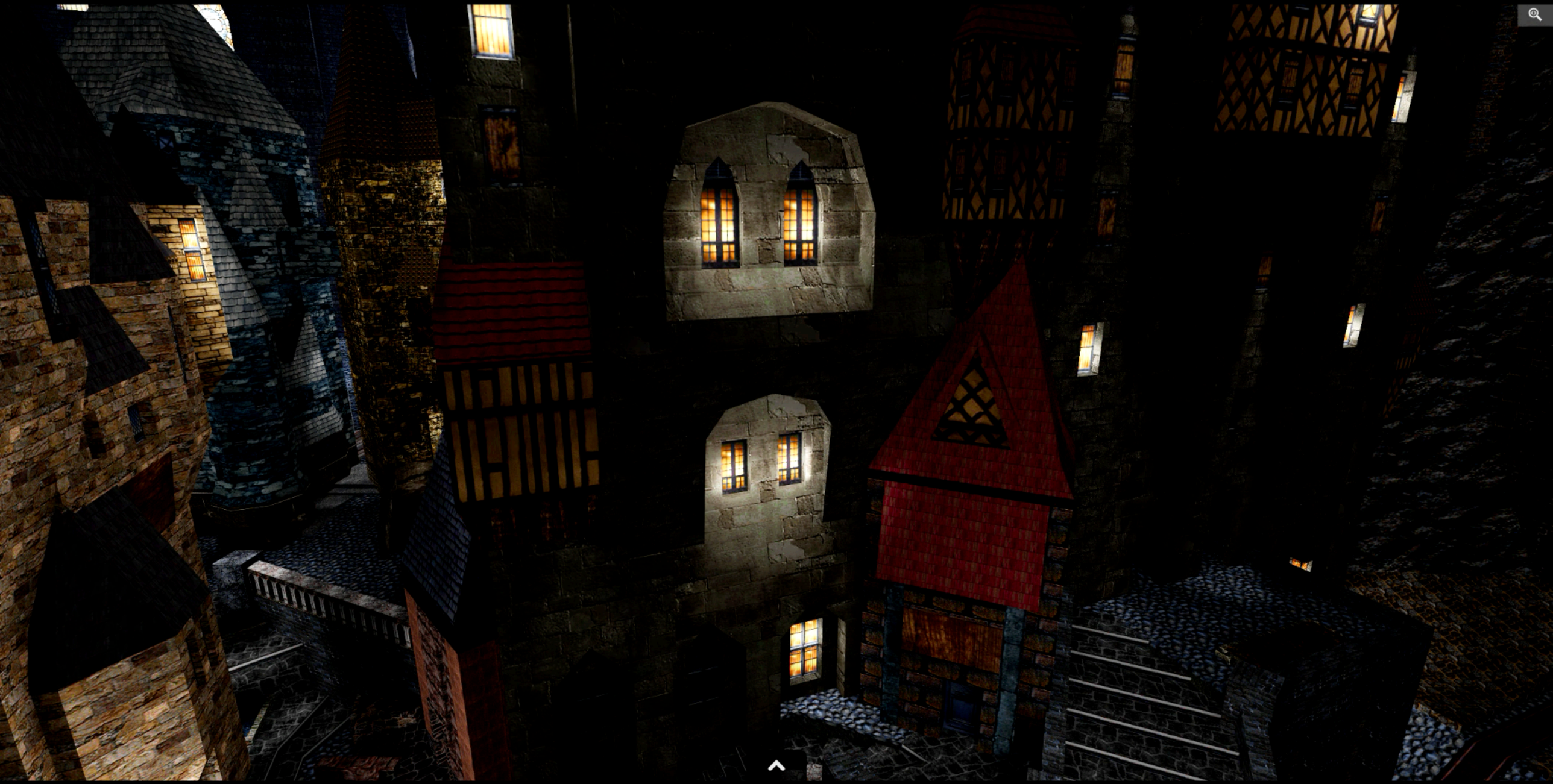

Lighting prototype

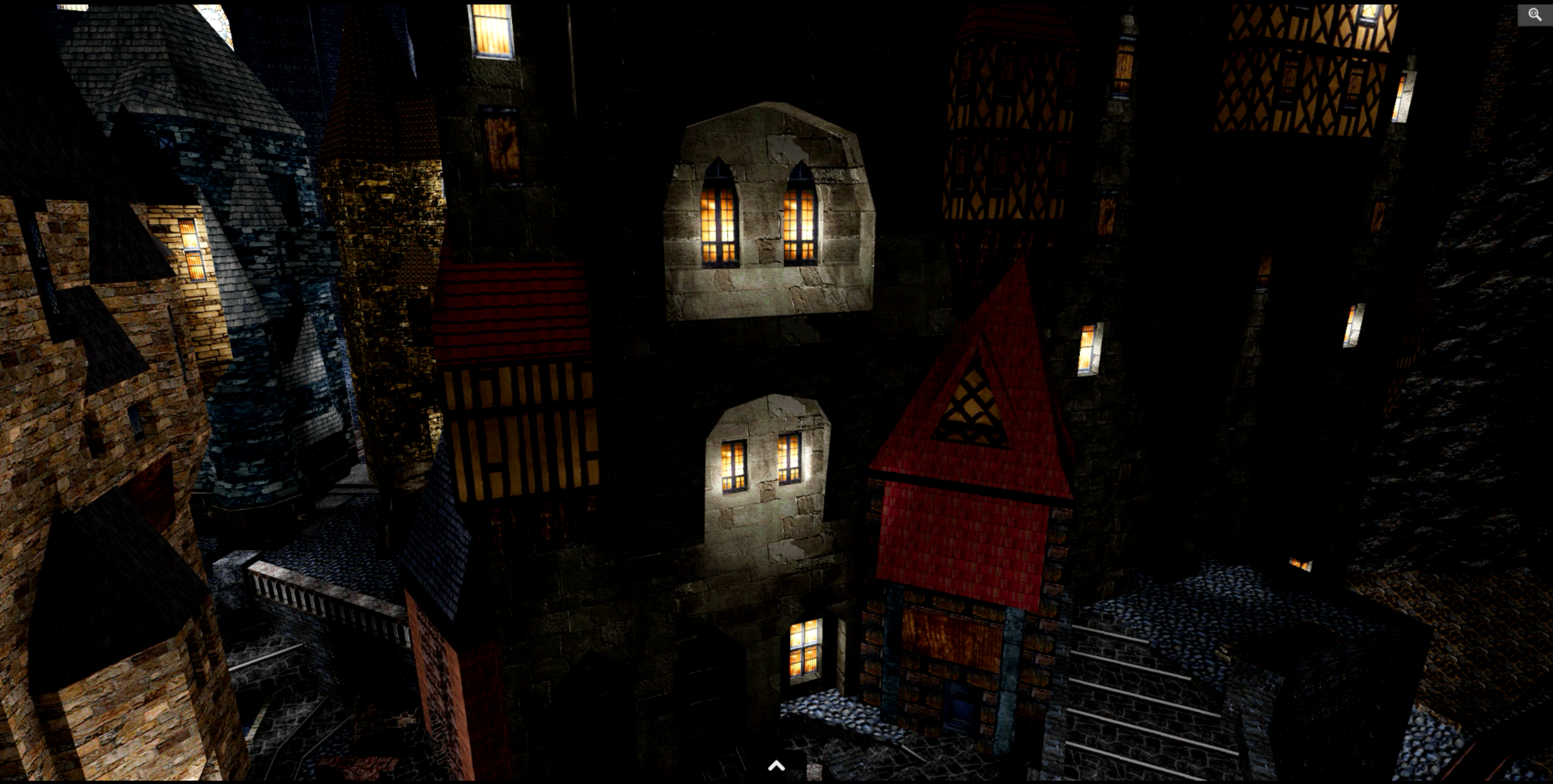

Lighting prototype

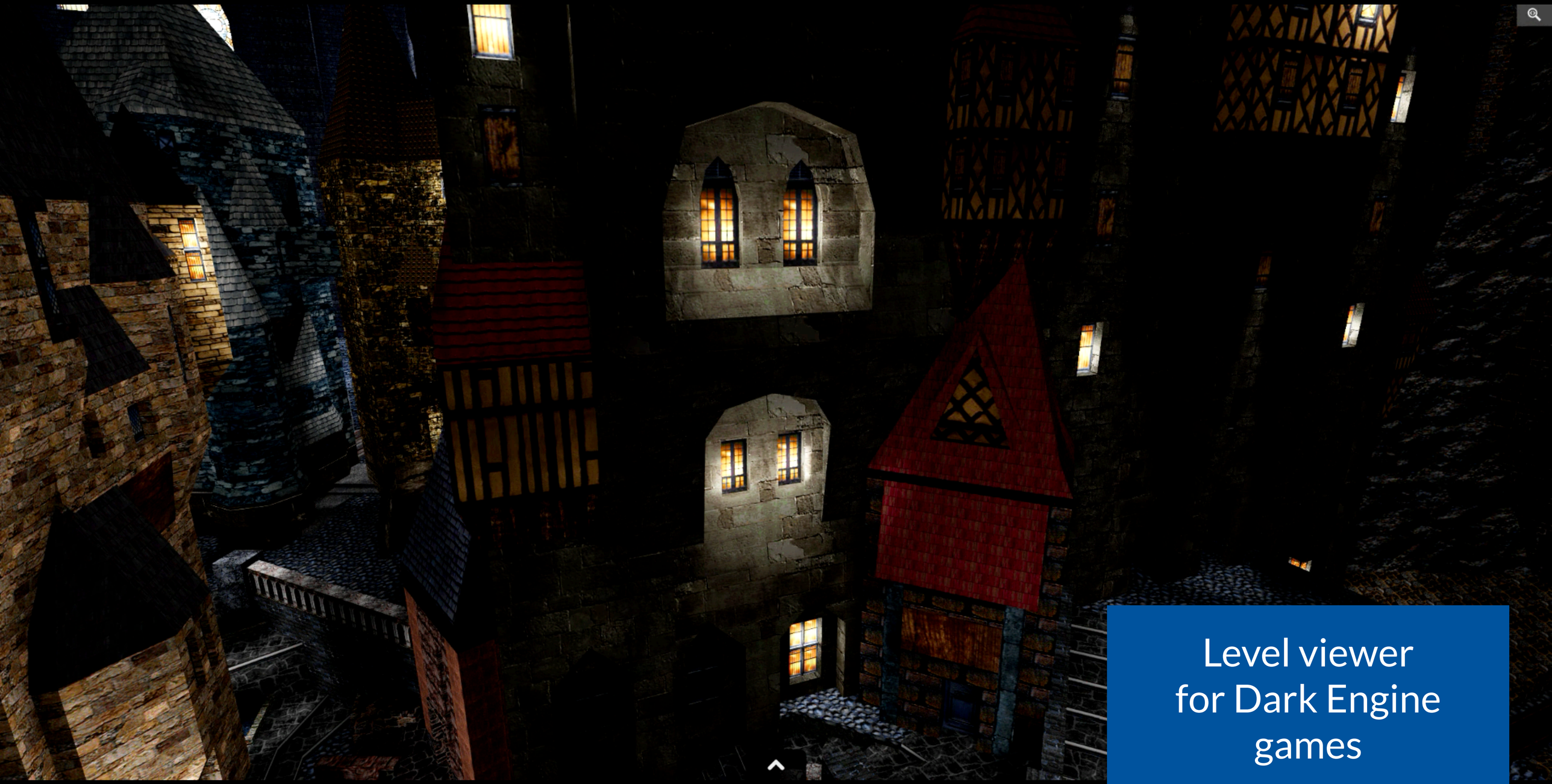

printf(…) rays
Lighting prototype


<LineLayer>
<PointLayer>
printf(…) rays
Lighting prototype


@link fn emitLine(…)
@link fn emitPoint(…)
<LineLayer>
<PointLayer>
printf(…) rays
Lighting prototype






Level viewer
for Dark Engine
games

Use.GPU works* today

Use.GPU works* today

*WebGPU support

Use.GPU works* today

*WebGPU support

But Use.GPU is a *what if*

Use.GPU works* today

*WebGPU support

But Use.GPU is a *what if*

Vertical Slice

Use.GPU works* today

*WebGPU support

But Use.GPU is a *what if*

Components

Vertical Slice

Components

Use.GPU works* today

*WebGPU support

But Use.GPU is a *what if*

Components

Vertical Slice

Components

Because the GPU API is **in the way**

Use.GPU works* today

*WebGPU support

But Use.GPU is a *what if*

Components

Vertical Slice

Components

GPGPU = **Dark Souls**

Because the GPU API is **in the way**

Use.GPU works* today

*WebGPU support

But Use.GPU is a *what if*

Only 4
equipment slots

Vertical
Slice

Components

GPGPU = **Dark Souls**

Because the GPU API
is **in the way**

Use.GPU works* today

*WebGPU support

But Use.GPU is a *what if*

Only 4
equipment slots

Vertical
Slice

Components

GPGPU = **Dark Souls**

Because the GPU API
is **in the way**

Industry is in **denial**

Use.GPU works* today

*WebGPU support

But Use.GPU is a *what if*

Only 4
equipment slots

Vertical
Slice

Multi-threading
in browsers
also sucks

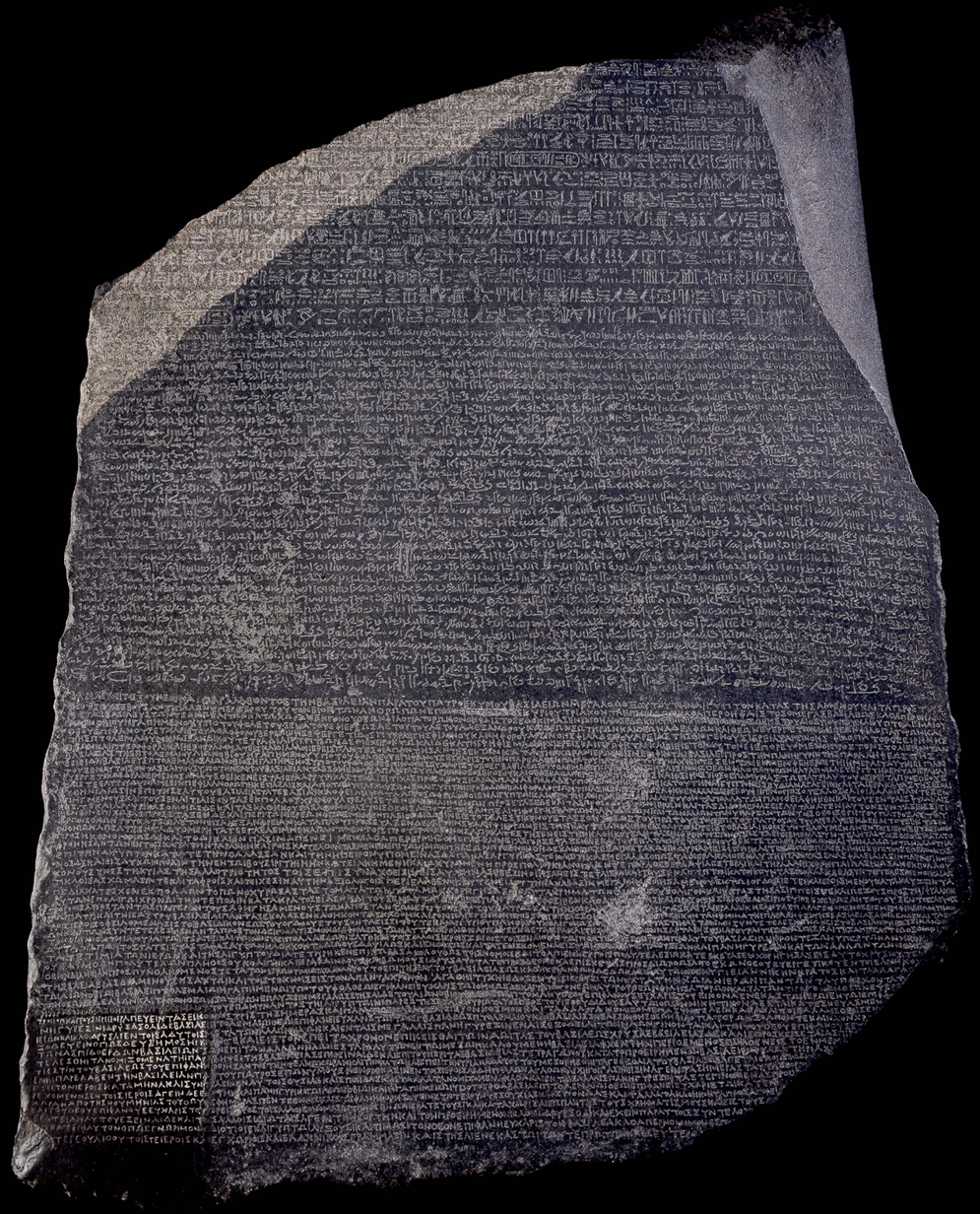
GPGPU = **Dark Souls**

Because the GPU API
is **in the way**

Industry is in **denial**

Incremental Effects

- OrbitCamera
- Provide(FrameContext)
- ViewProvider
  - Signal
  - Provide(ViewContext)
  - Provide(LayoutContext)
    - Memo(Cursor)
    - Memo(Pass)
    - Memo(ForwardRenderer)
    - Reconcile
      - Quote
        - Provide(VariantContext)
          - Data
            - Signal
            - Memo(RawLines)
          - Data
            - Signal
            - Memo(RawLines)
          - Data
            - Signal
            - Memo(ArrowLayer)
              - Memo(RawLines)
              - Memo(RawArrows)
      - Resolve(Quote)
    - Root(PassReconciler)
      - Provide(PassContext)
      - MultiGather
        - Unquote
          - SolidRender
          - SolidRender
          - SolidRender
          - SolidRender
        - Resolve(Unquote)



- OrbitCamera
  - Provide(FrameContext)
    - ViewProvider
      - Signal
        - Provide(ViewContext)
          - Provide(LayoutContext)
            - Memo(Cursor)
              - Memo(Pass)
                - Memo(ForwardRenderer)
                  - Reconcile
                    - Quote
                      - Provide(VariantContext)
                        - Data
                          - Signal
                          - Memo(RawLines)
                        - Data
                          - Signal
                          - Memo(RawLines)
                        - Data
                          - Signal
                          - Memo(ArrowLayer)
                            - Memo(RawLines)
                            - Memo(RawArrows)
                    - Resolve(Quote)
                  - Root(PassReconciler)
                    - Provide(PassContext)
                    - MultiGather
                      - Unquote
                        - SolidRender
                        - SolidRender
                        - SolidRender
                        - SolidRender
                      - Resolve(Unquote)

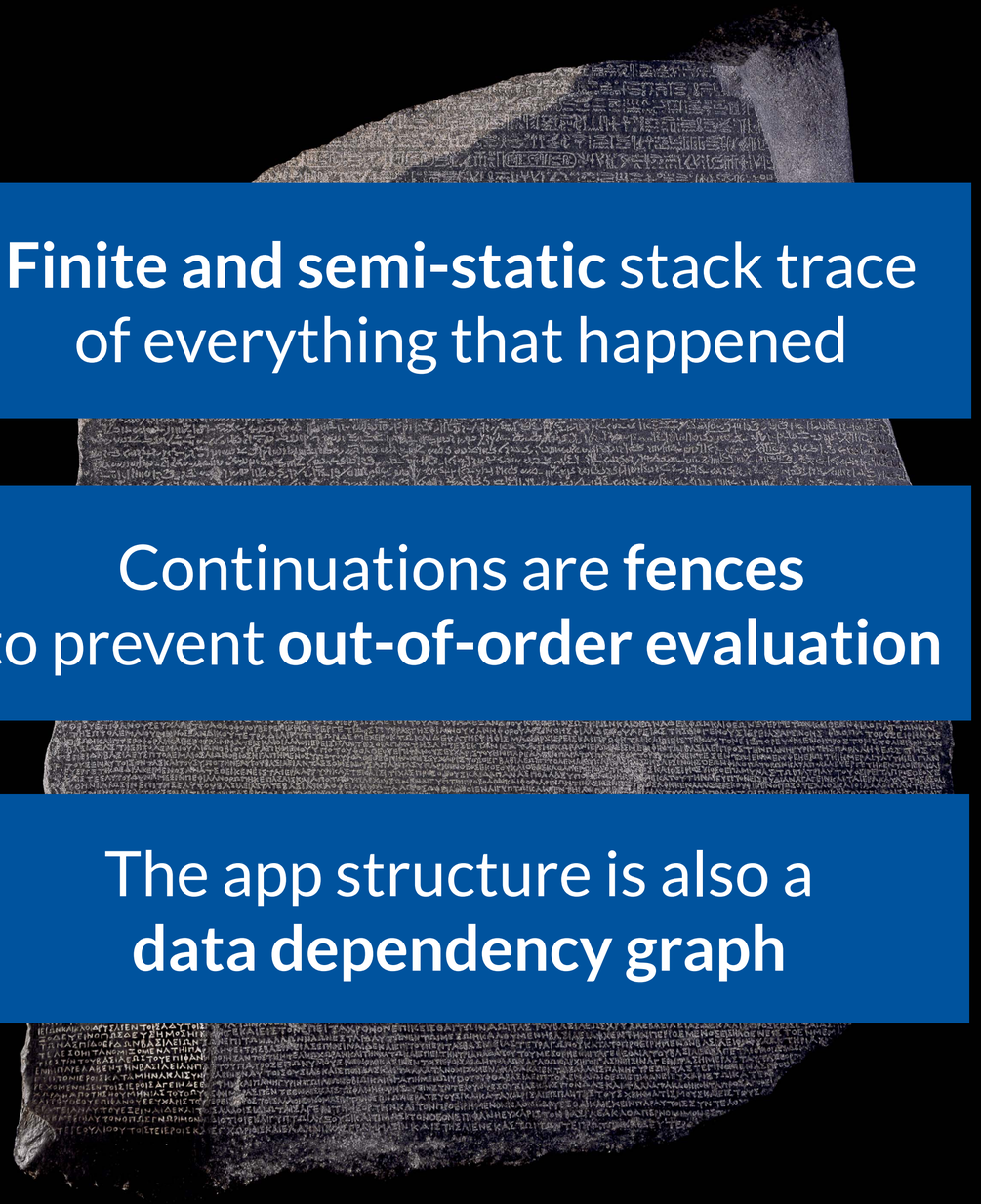
Finite and semi-static stack trace of everything that happened

- OrbitCamera
- Provide(FrameContext)
- ViewProvider
  - Signal
  - Provide(ViewContext)
  - Provide(LayoutContext)
    - Memo(Cursor)
    - Memo(Pass)
    - Memo(ForwardRenderer)
    - Reconcile
      - Quote
        - Provide(VariantContext)
          - Data
            - Signal
            - Memo(RawLines)
          - Data
            - Signal
            - Memo(RawLines)
          - Data
            - Signal
            - Memo(ArrowLayer)
              - Memo(RawLines)
              - Memo(RawArrows)
      - Resolve(Quote)
    - Root(PassReconciler)
      - Provide(PassContext)
      - MultiGather
        - Unquote
          - SolidRender
          - SolidRender
          - SolidRender
          - SolidRender
        - Resolve(Unquote)

Finite and semi-static stack trace of everything that happened

Continuations are **fences** to prevent **out-of-order evaluation**

- OrbitCamera
  - Provide(FrameContext)
    - ViewProvider
      - Signal
        - Provide(ViewContext)
          - Provide(LayoutContext)
            - Memo(Cursor)
              - Memo(Pass)
                - Memo(ForwardRenderer)
                  - Reconcile
                    - Quote
                      - Provide(VariantContext)
                        - Data
                          - Signal
                          - Memo(RawLines)
                        - Data
                          - Signal
                          - Memo(RawLines)
                        - Data
                          - Signal
                          - Memo(ArrowLayer)
                            - Memo(RawLines)
                            - Memo(RawArrows)
                    - Resolve(Quote)
                  - Root(PassReconciler)
                    - Provide(PassContext)
                    - MultiGather
                      - Unquote
                        - SolidRender
                        - SolidRender
                        - SolidRender
                        - SolidRender
                      - Resolve(Unquote)

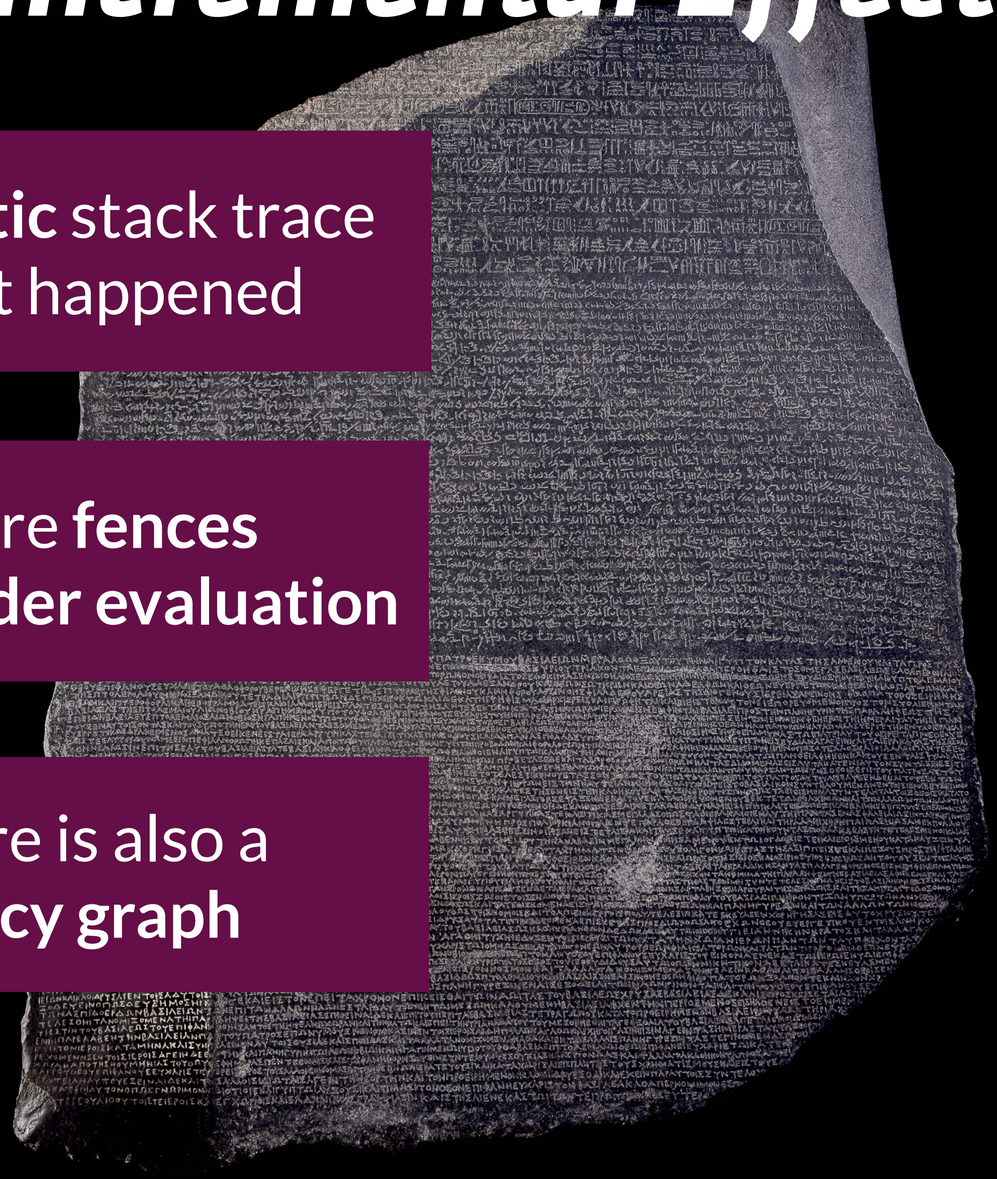


Finite and semi-static stack trace of everything that happened

Continuations are **fences** to prevent **out-of-order evaluation**

The app structure is also a **data dependency graph**

Incremental Effects

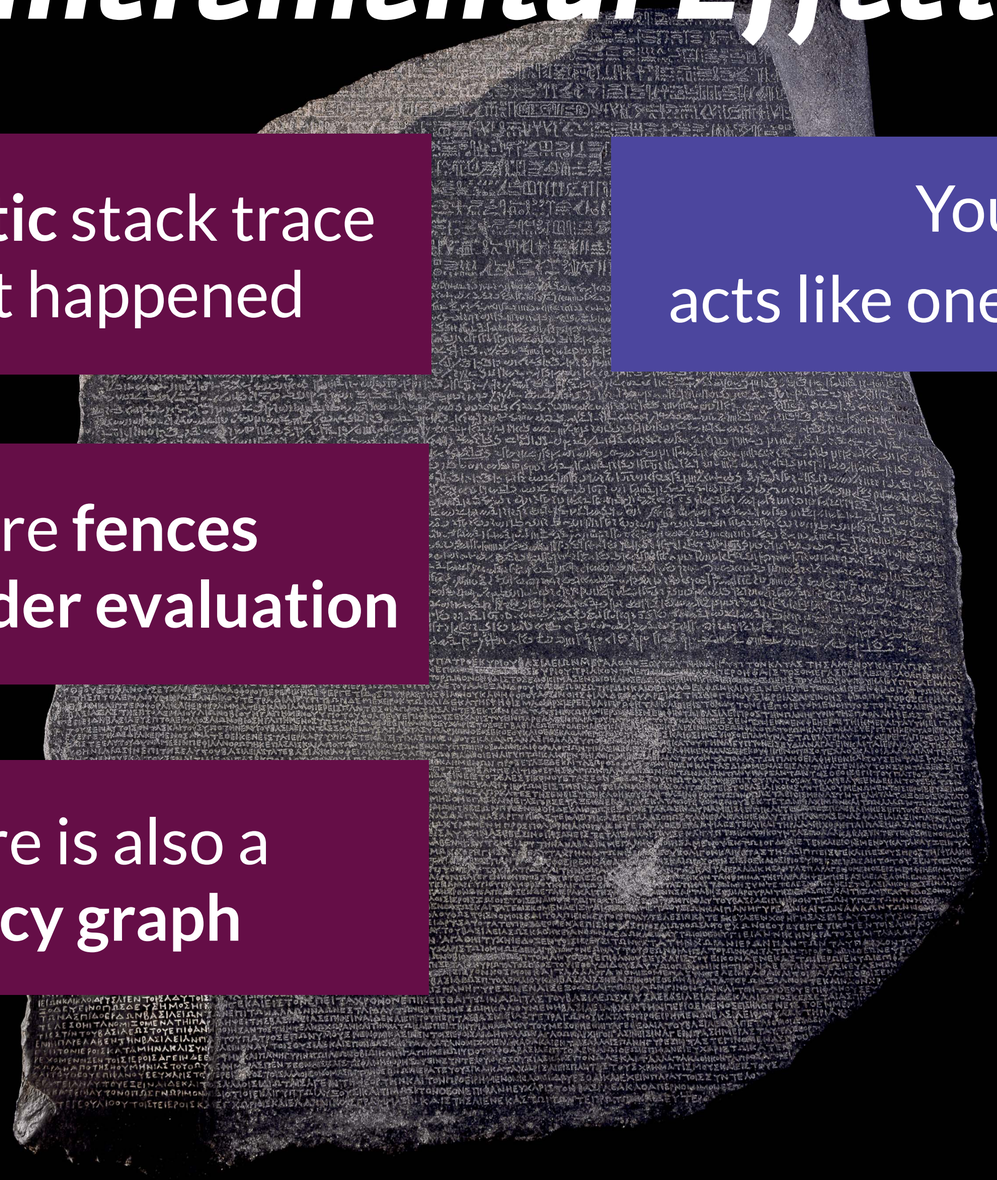


Finite and semi-static stack trace
of everything that happened

Continuations are **fences**
to prevent **out-of-order evaluation**

The app structure is also a
data dependency graph

Incremental Effects



Finite and semi-static stack trace
of everything that happened

Your **entire app**
acts like one big **`main()`** function

Continuations are **fences**
to prevent **out-of-order evaluation**

The app structure is also a
data dependency graph

Incremental Effects

Finite and semi-static stack trace
of everything that happened

Your **entire app**
acts like one big **`main()`** function

Continuations are **fences**
to prevent **out-of-order evaluation**

The app structure is also a
data dependency graph

Incremental Effects

Finite and semi-static stack trace
of everything that happened

Your **entire app**
acts like one big **`main()`** function

Continuations are **fences**
to prevent **out-of-order evaluation**

DSL for the **target data structure**

The app structure is also a
data dependency graph

Incremental Effects

Finite and semi-static stack trace
of everything that happened

Your **entire app**
acts like one big **`main()`** function

Continuations are **fences**
to prevent **out-of-order evaluation**

DSL for the **target data structure**

The app structure is also a
data dependency graph

Hierarchy of change
ordered by granularity

Incremental Effects

Finite and semi-static stack trace
of everything that happened

Your **entire app**
acts like one big **`main()`** function

Continuations are **fences**
to prevent **out-of-order evaluation**

DSL for the **target data structure**

The app structure is also a
data dependency graph

Hierarchy of change
ordered by granularity

CPU code acts like a JITed **GPU**

Incremental Effects

Finite and semi-static stack trace
of everything that happened

Your **entire app**
acts like one big **`main()`** function

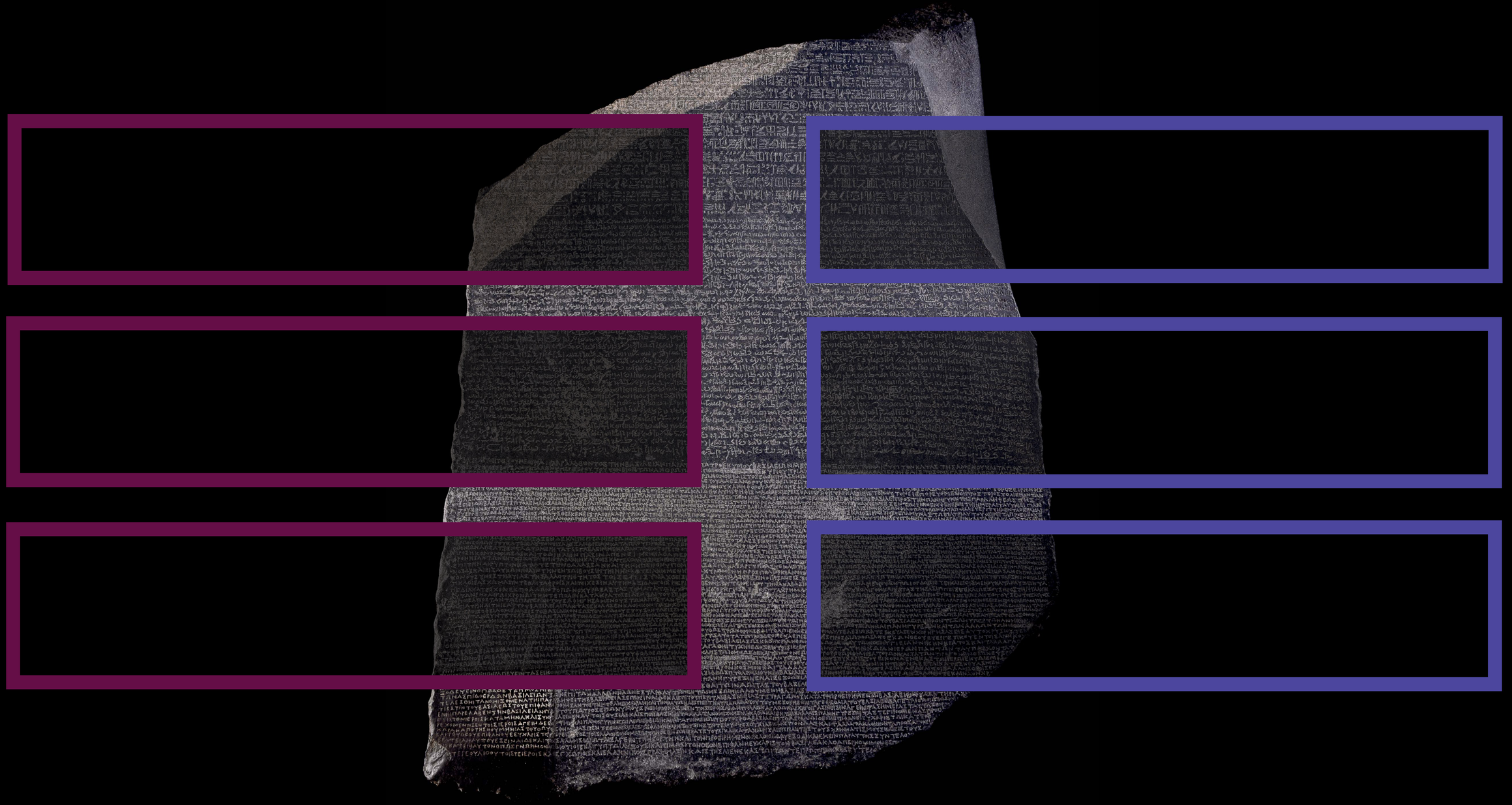
Continuations are **fences**
to prevent **out-of-order evaluation**

DSL for the **target data structure**

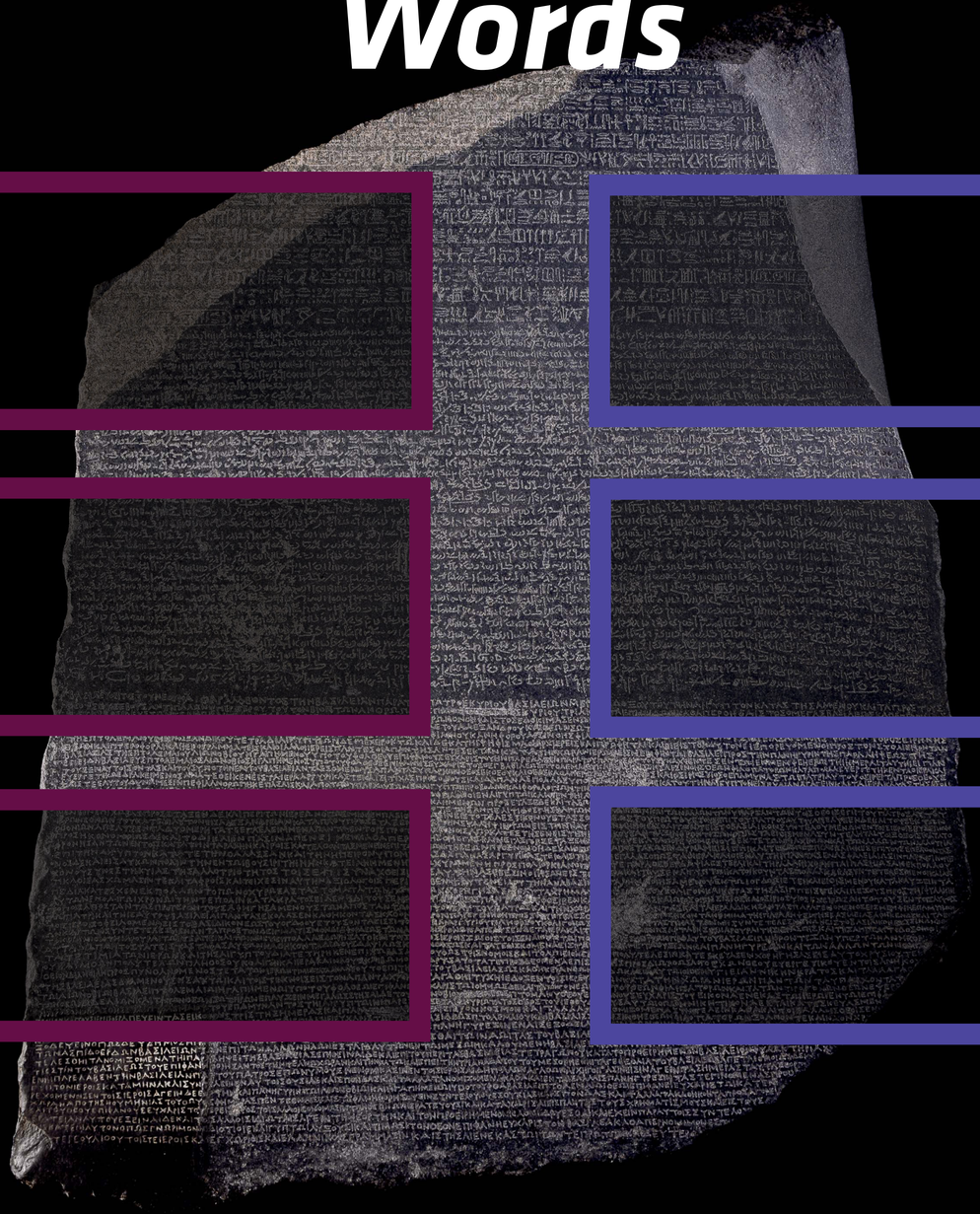
The app structure is also a
data dependency graph

Hierarchy of change
ordered by granularity

CPU code acts like **immediate mode**



Words



Words

Finite and semi-static log
of everything that happened

Words

Finite and semi-static log
of everything that happened

Your **entire story**
acts like one big **`main()`** function

Words

Finite and semi-static log
of everything that happened

Your **entire story**
acts like one big **`main()`** function

Words

Finite and semi-static log
of everything that happened

Your **entire story**
acts like one big **`main()`** function

Sentences are **fences**
to prevent out-of-order evaluation

Words

Finite and semi-static log
of everything that happened

Your **entire story**
acts like one big **`main()`** function

Sentences are **fences**
to prevent out-of-order evaluation

DSL for the **target world**

Words

Finite and semi-static log
of everything that happened

Your **entire story**
acts like one big `main()` function

Sentences are **fences**
to prevent out-of-order evaluation

DSL for the **target world**

The **story structure** is also a
data dependency graph

Words

Finite and semi-static log
of everything that happened

Your **entire story**
acts like one big `main()` function

Sentences are **fences**
to prevent out-of-order evaluation

DSL for the **target world**

The **story structure** is also a
data dependency graph

Hierarchy of change
ordered by granularity

Words

Finite and semi-static log
of everything that happened

Your **entire story**
acts like one big `main()` function

Sentences are **fences**
to prevent out-of-order evaluation

DSL for the **target world**

The **story structure** is also a
data dependency graph

Hierarchy of change
ordered by granularity

LLM acts like a JITed **world**

Words

Finite and semi-static log
of everything that happened

Your **entire story**
acts like one big **`main()`** function

Sentences are **fences**
to prevent out-of-order evaluation

DSL for the **target world**

The **story structure** is also a
data dependency graph

Hierarchy of change
ordered by granularity

A language is an effect system
that **orders information** by how **often** context changes

Words

Finite and semi-static log
of everything that happened

Your **entire story**
acts like one big `main()` function

Sentences are **fences**
to prevent out-of-order evaluation

DSL for the **target world**

The **story structure** is also a
data dependency graph

Hierarchy of change
ordered by granularity

A language is an effect system
that **orders information** by how **often** context changes

Zipf's Law 2.0

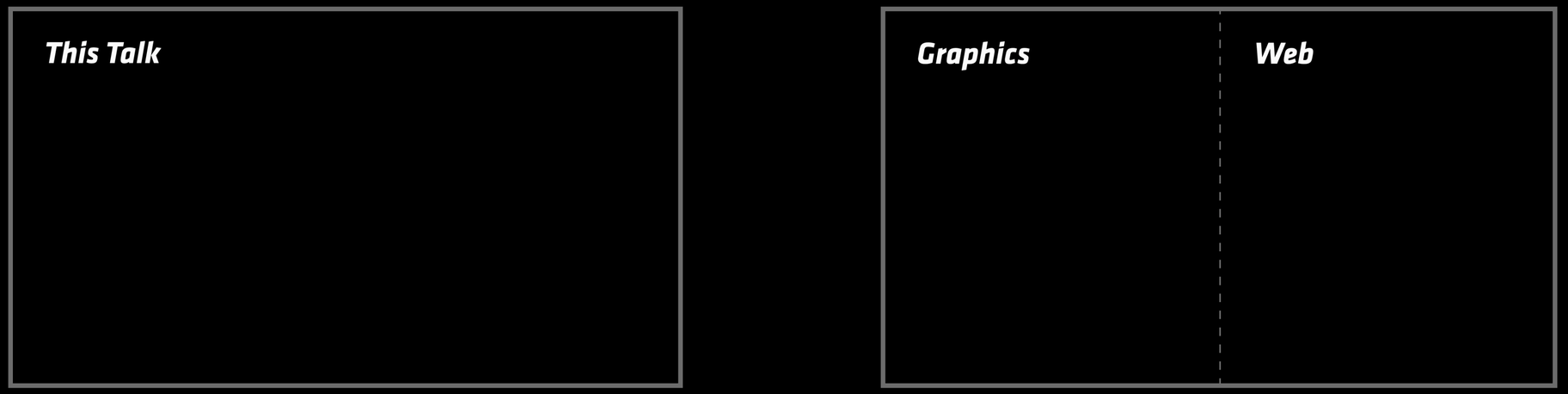
Shapes

Shapes

Shapes

Mounting components

Shapes



Mounting components

Shapes

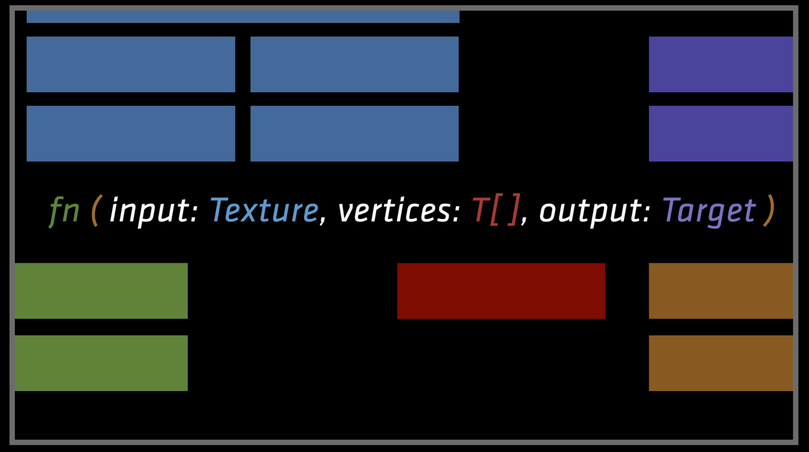
Mounting components

Reconciling trees

Shapes

Mounting components

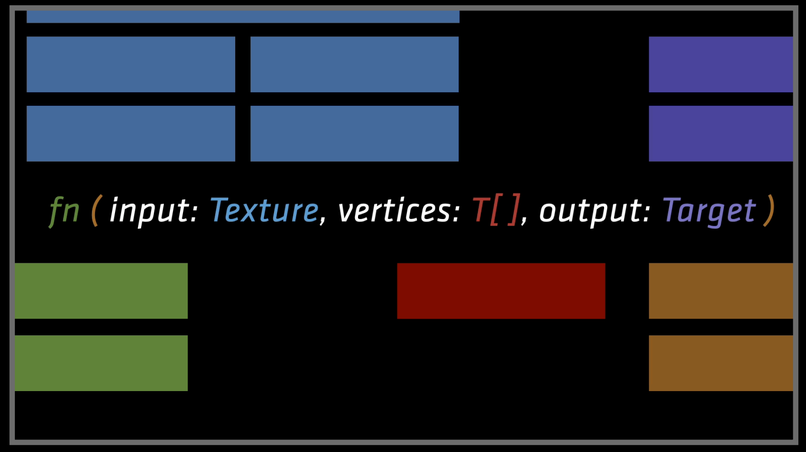
Reconciling trees



Shapes

Mounting components

Reconciling trees



Reducing information

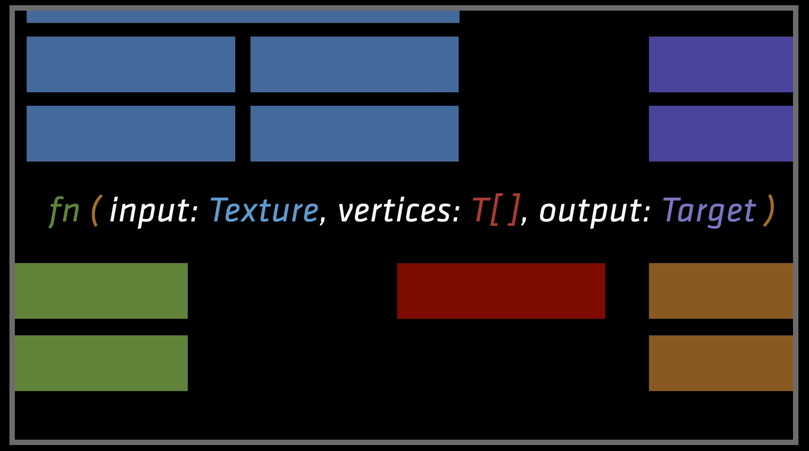
Shapes

This Talk

Mounting components

Graphics ***Web***

Reconciling trees



Reducing information

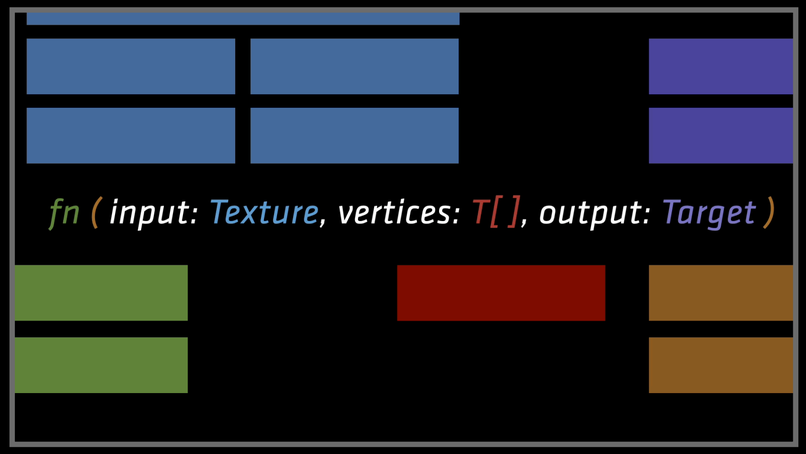
"How do I draw a f😾king line"

Graphics ***Web***

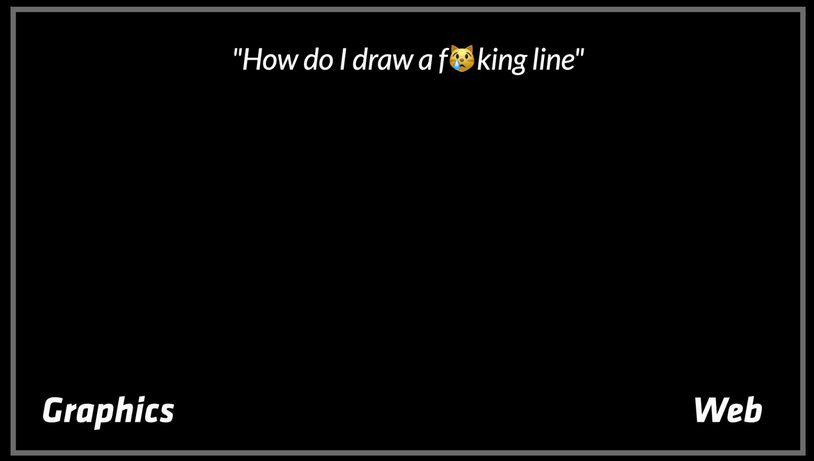
Shapes

Mounting components

Reconciling trees



Reducing information

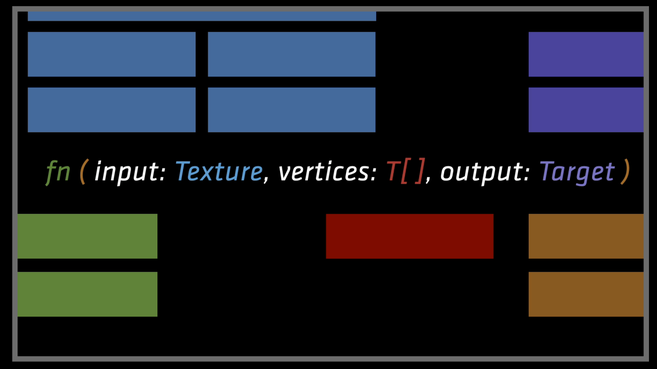


Quoting

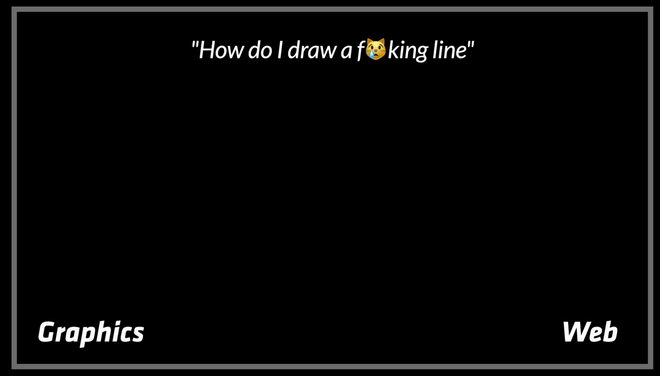
Shapes

Mounting components

Reconciling trees



Reducing information



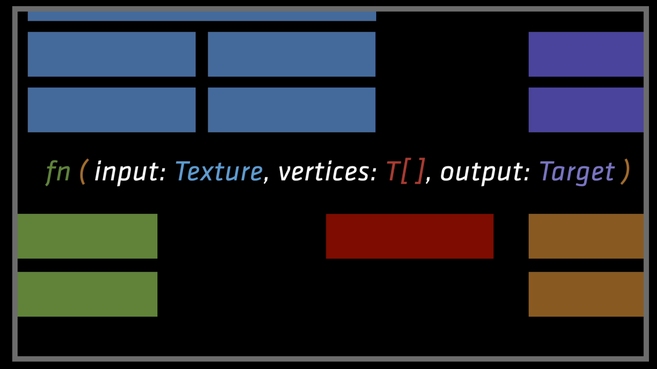
Quoting

Quoting

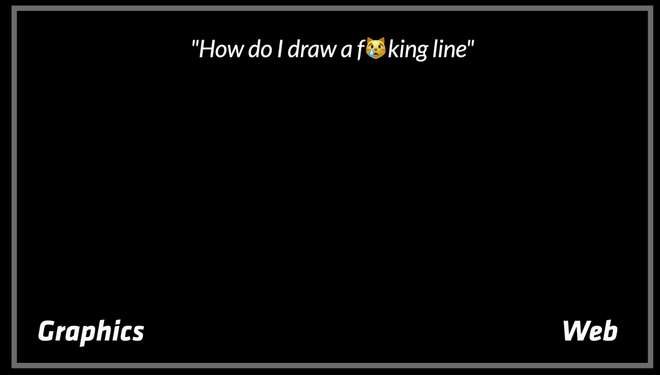
Shapes

Calling functions

Reconciling trees



Reducing information



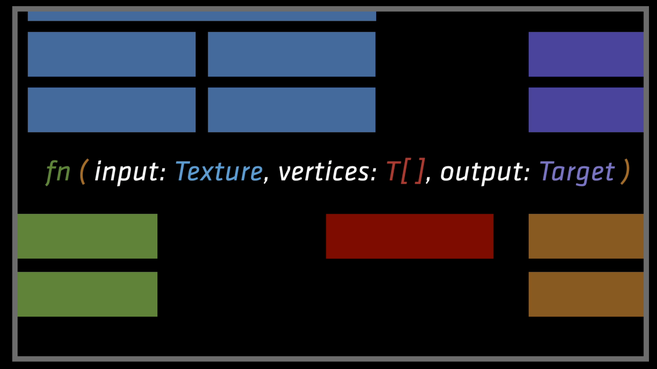
Quoting

Quoting

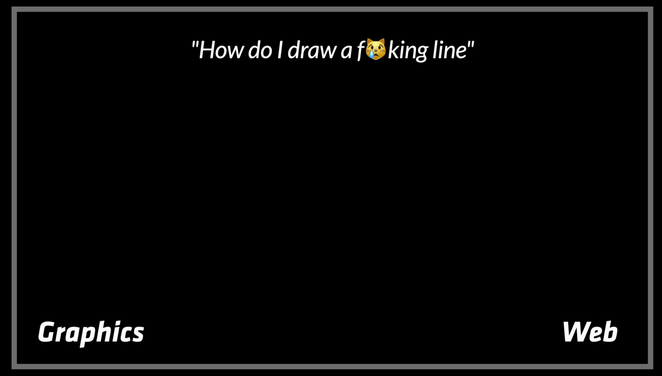
Shapes

Calling functions

Diffing maps



Reducing information



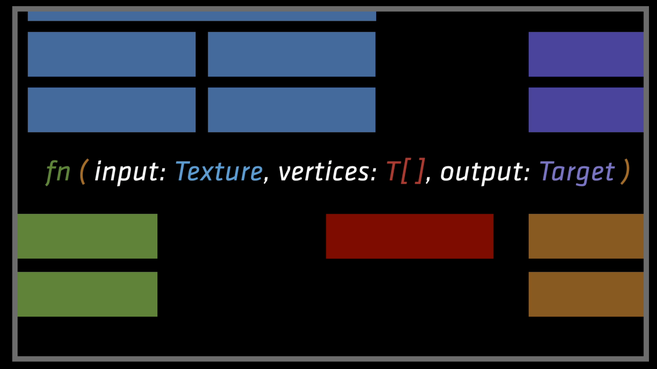
Quoting

Quoting

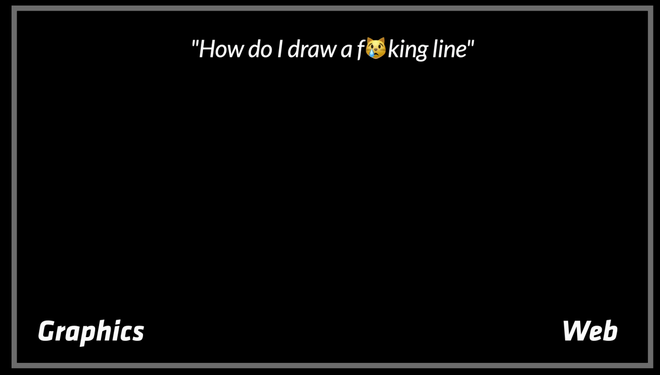
Shapes

Calling functions

Diffing maps



Returning values



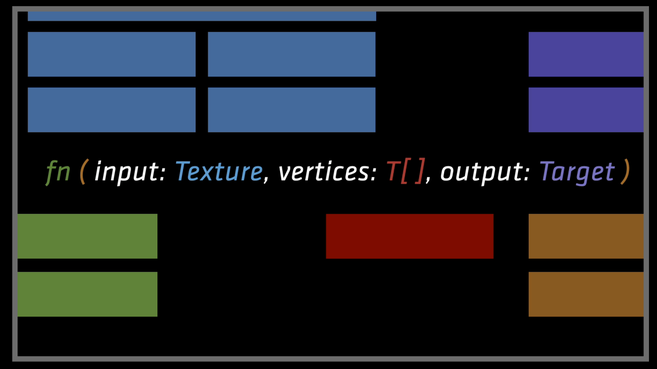
Quoting

Quoting

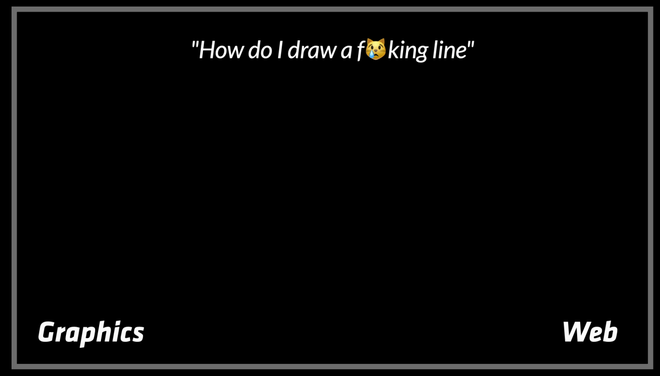
Shapes

Calling functions

Diffing maps



Returning values



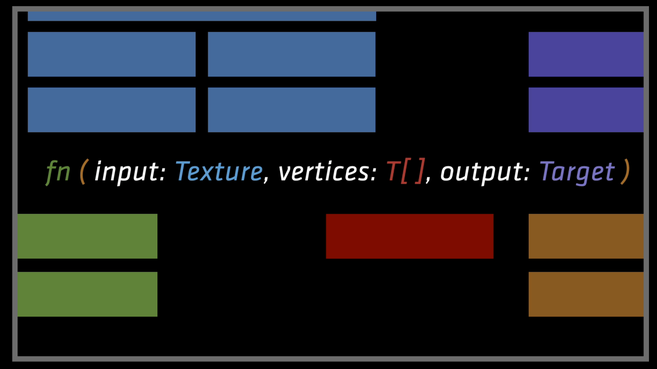
Sandboxing

Sandboxing

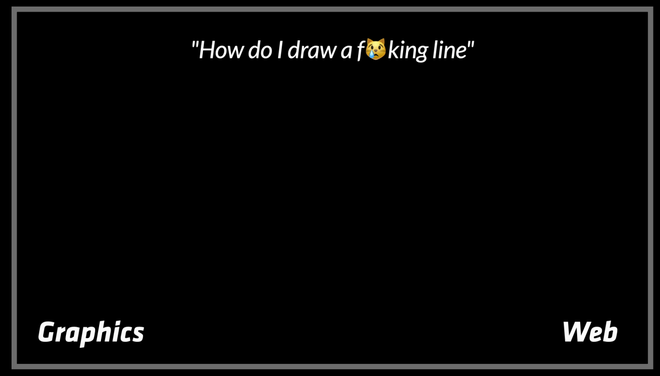
Shapes

Calling functions

Diffing maps



Returning values



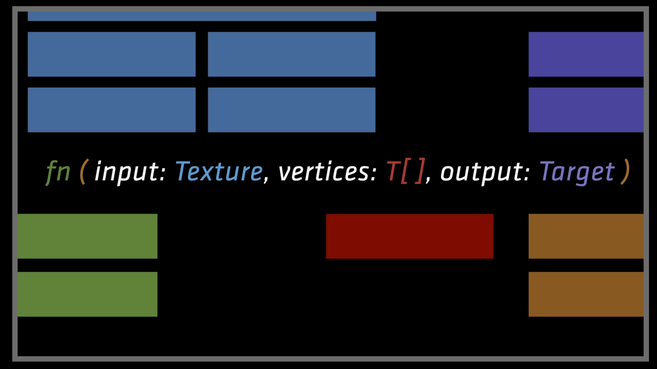
Sandboxing

Sandboxing

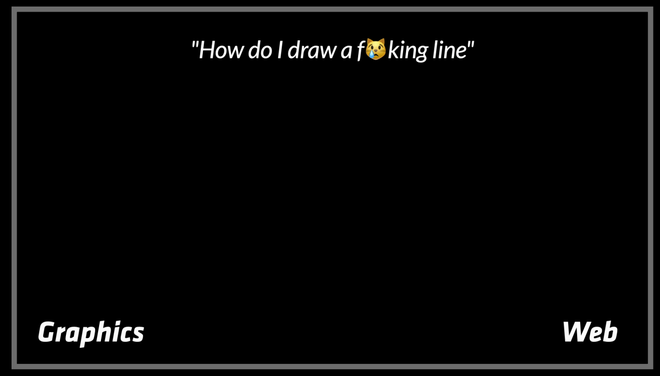
Shapes

Calling functions

Diffing maps



Returning values



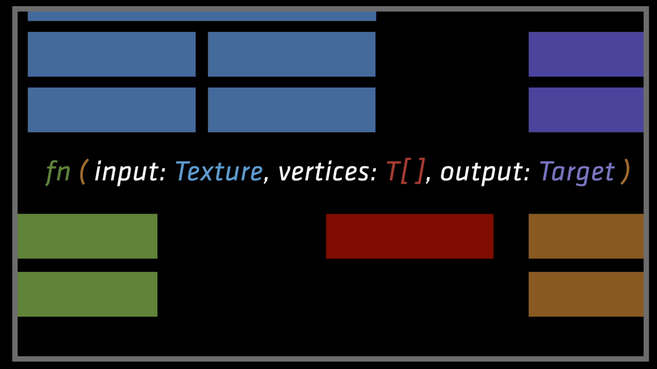
Sandboxing

Sandboxing

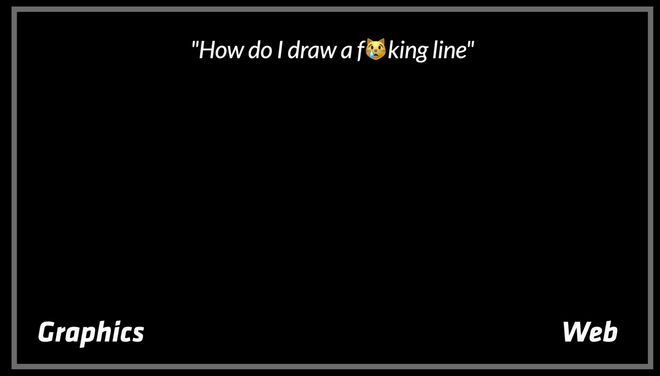
Shapes

Calling functions

Diffing maps



Returning values



Sandboxing

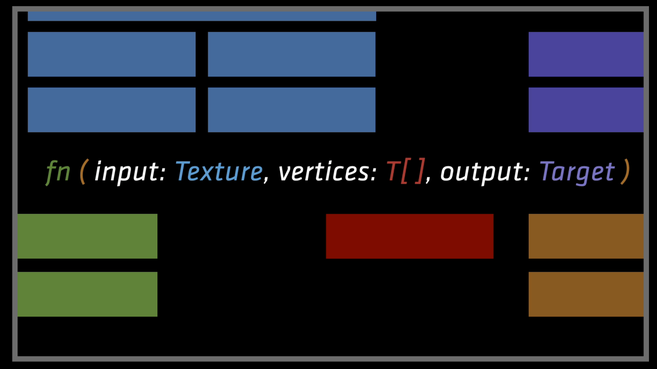
Sandboxing

Use.GPU

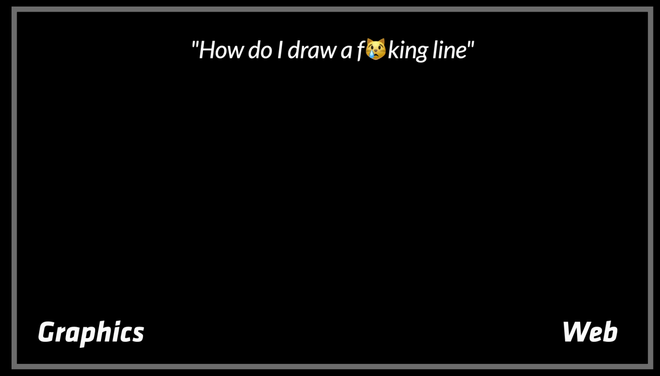
Shapes

Calling functions

Diffing maps



Returning values



Sandboxing

Sandboxing

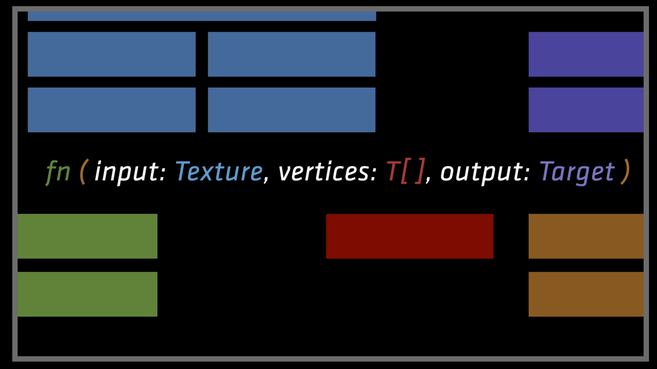
Use.GPU

//usegpu.live

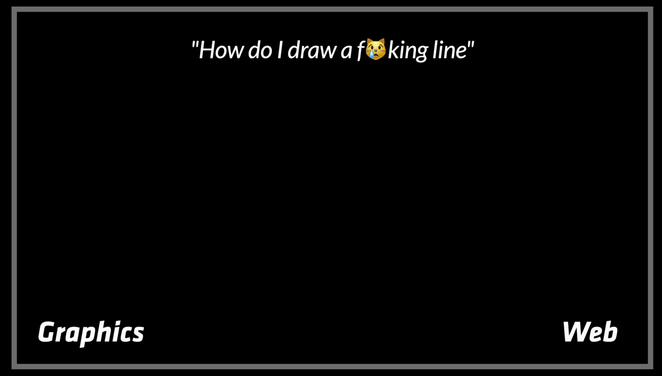
Shapes

Calling functions

Diffing maps



Returning values



Sandboxing

Sandboxing

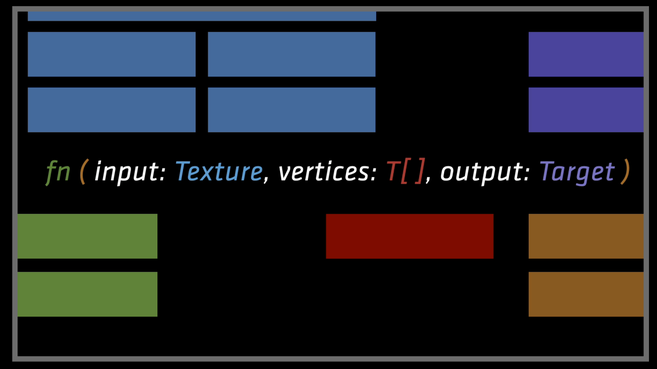
Use.GPU

//usegpu.live

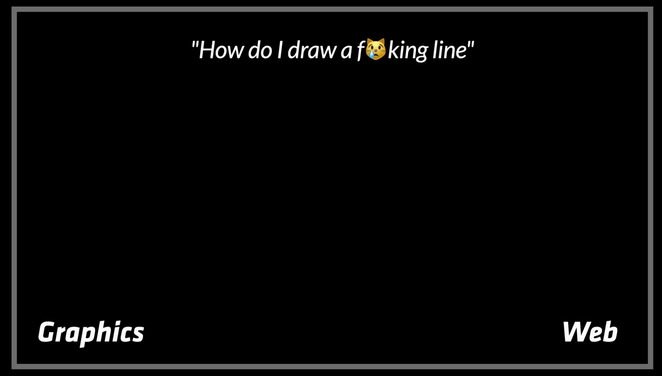
Shapes

Calling functions

Diffing maps



Returning values



Sandboxing

Sandboxing

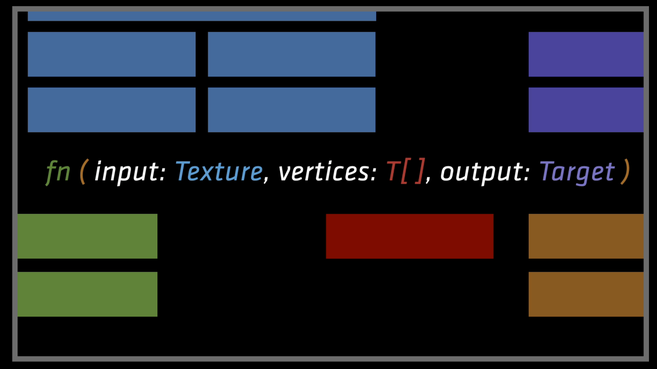
Use.GPU

//usegpu.live

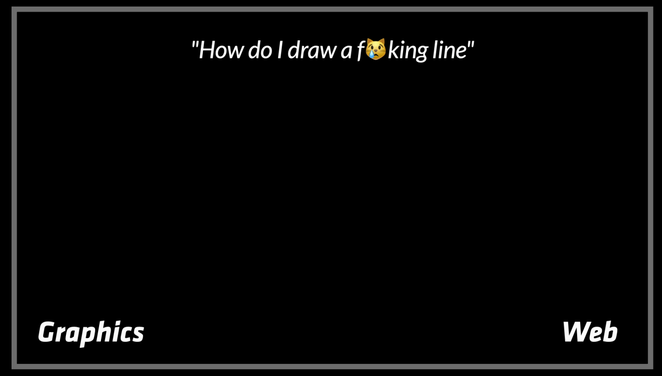
Shapes

Calling functions

Diffing maps



Returning values



Sandboxing

Sandboxing

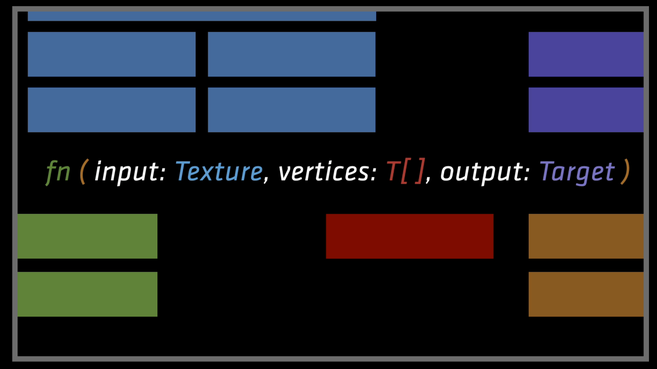
Use.GPU

//usegpu.live

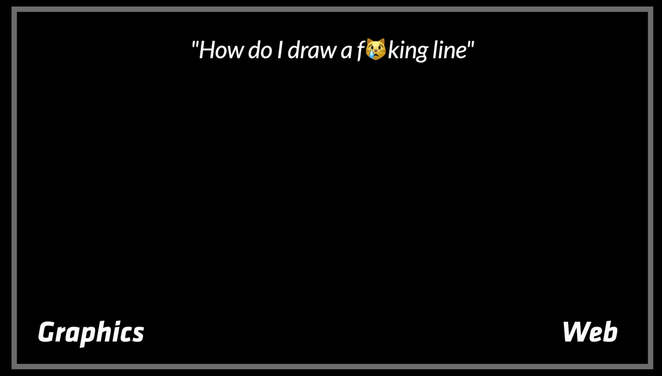
Shapes

Calling functions

Diffing maps



Returning values



Sandboxing

Sandboxing