Agenda

- ~~Introduction~~
- ~~JavaScriptCore~~
- ~~Efficient Mark-Sweep~~

  - *30 minute break*

- Concurrent GC
- bmalloc
- WTF::Lock

Riptide: WebKit's Concurrent Garbage Collector

Filip Pizlo
Apple

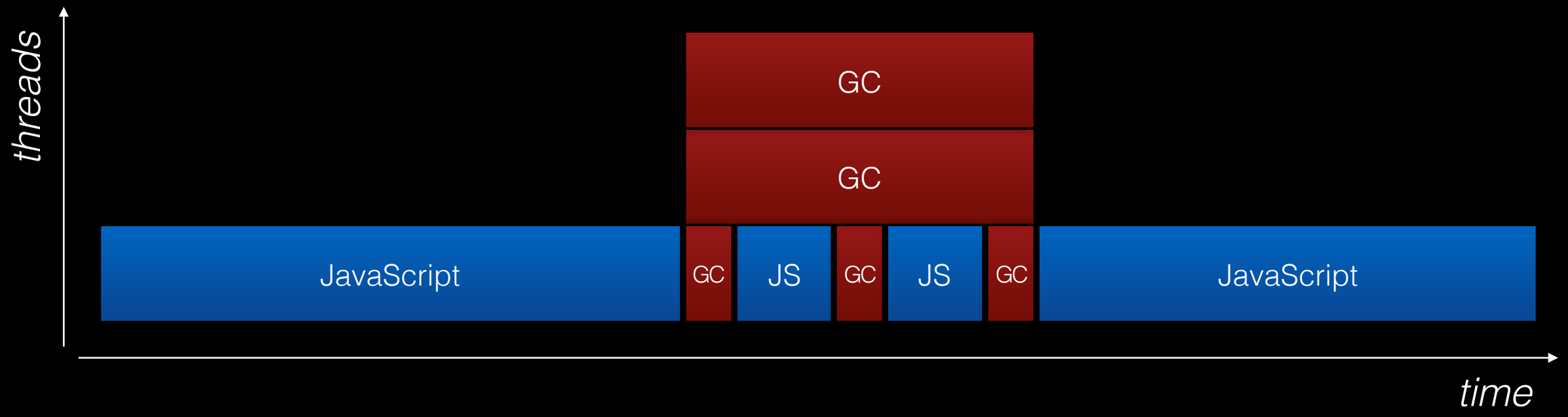
Riptide

- Introduction
- Retreating Wavefront
- Space-Time Scheduler
- Safepoints
- Obstruction-Free Double Collect Snapshot
- Embedded WTF Locks
- Conclusion

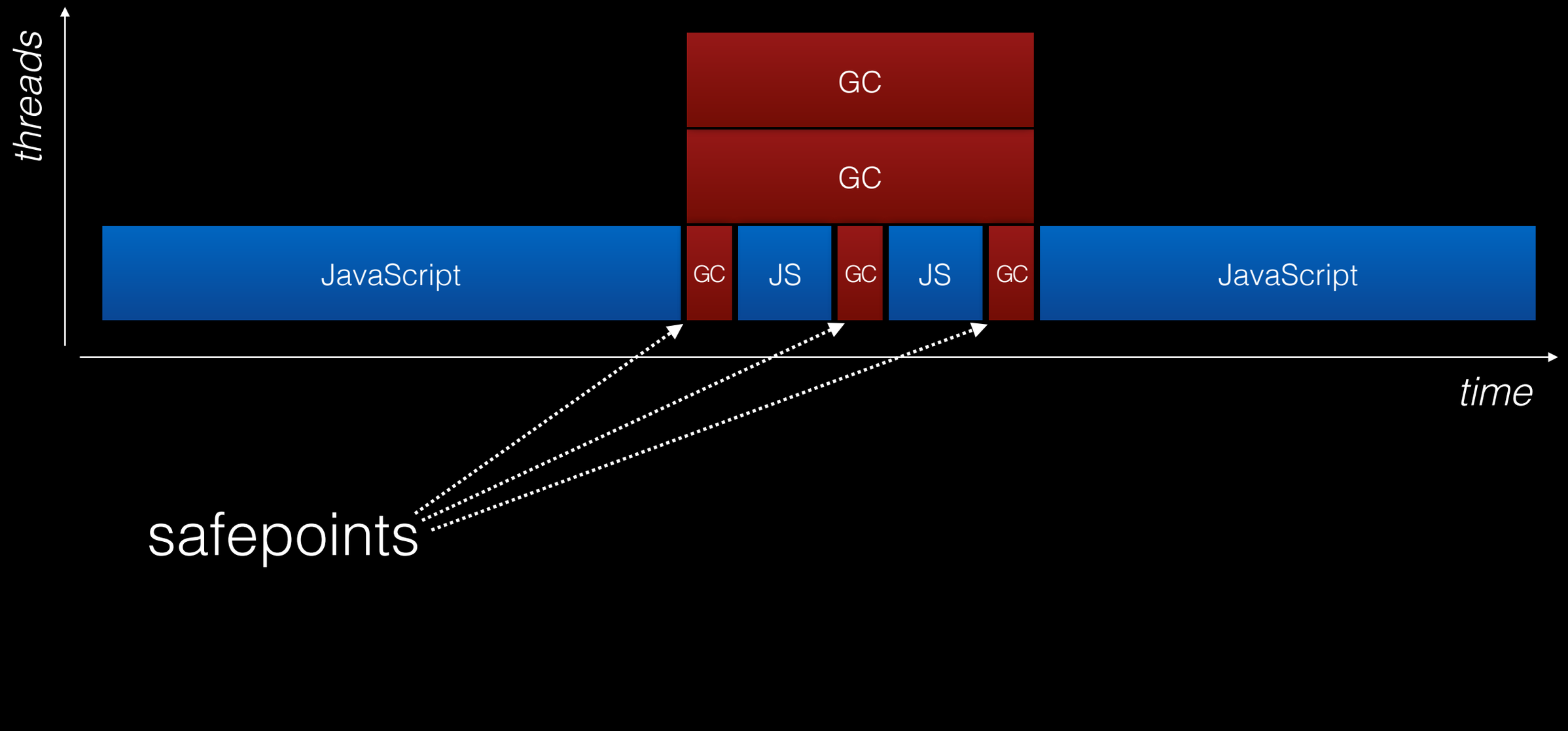
Riptide

- Introduction
- Retreating Wavefront
- Space-Time Scheduler
- Safepoints
- Obstruction-Free Double Collect Snapshot
- Embedded WTF Locks
- Conclusion

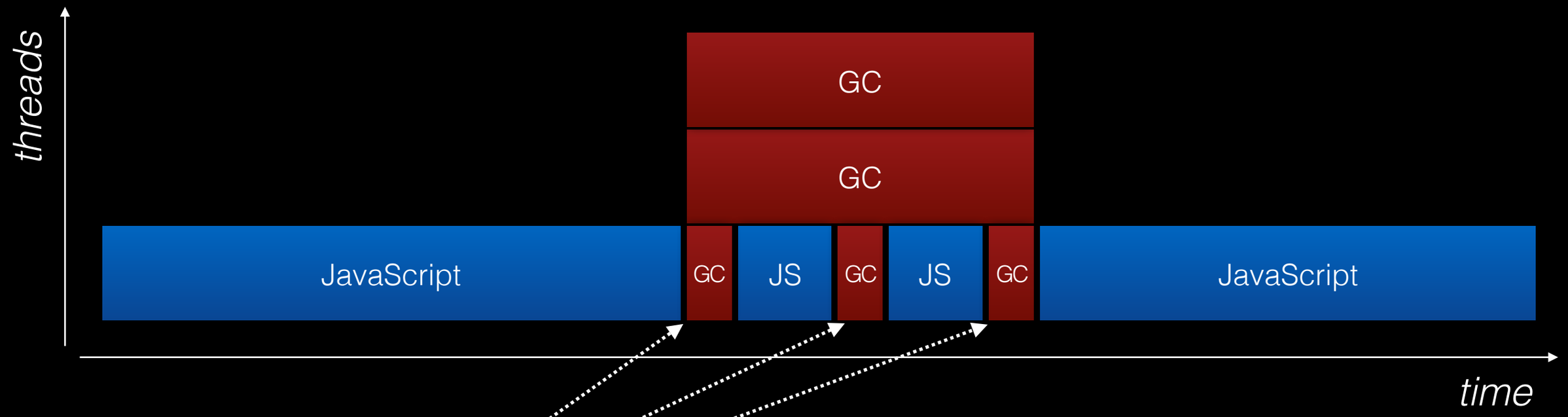
Concurrent GC



Concurrent GC



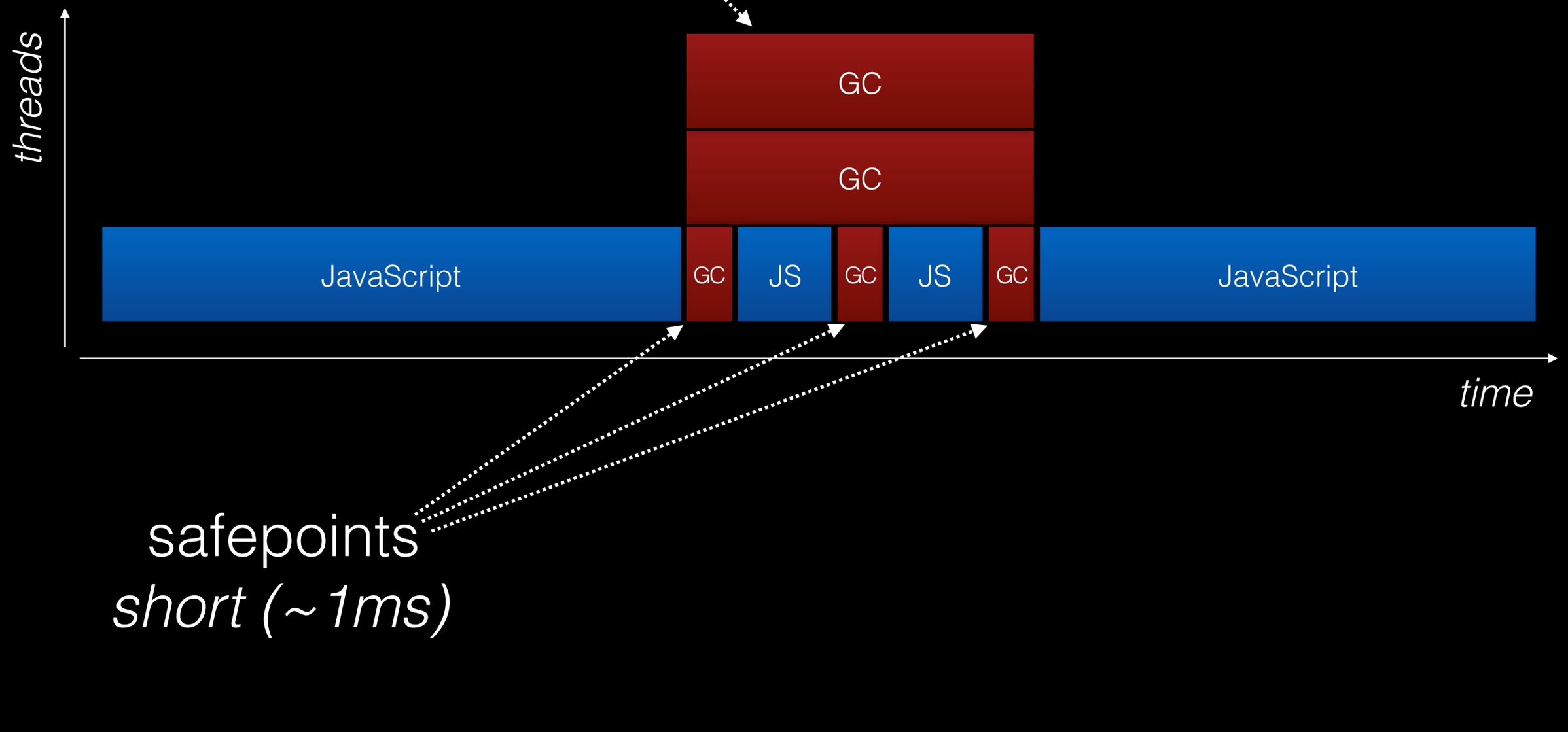
Concurrent GC



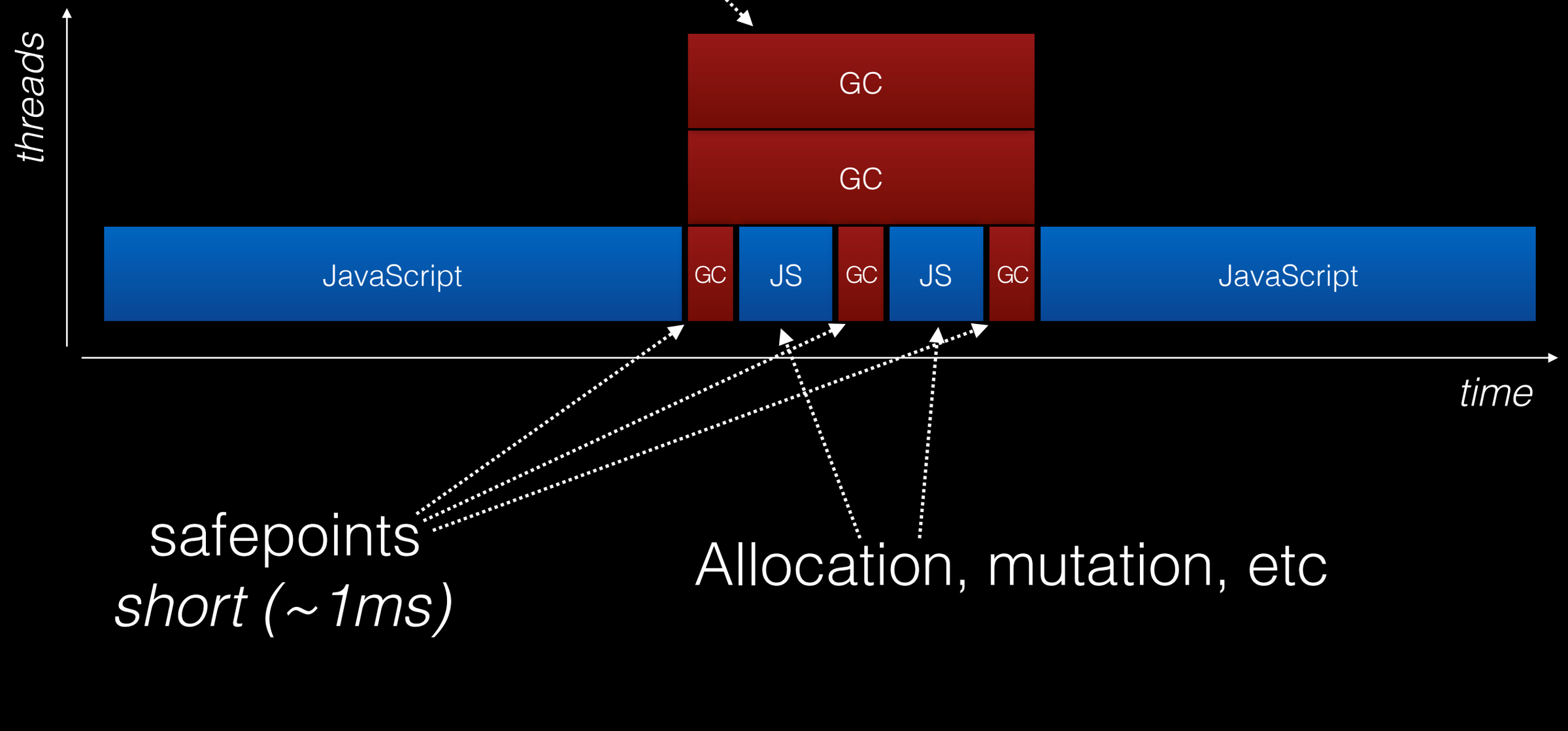
safepoints

short (~1ms)

Concurrent GC



Concurrent GC



Desired Outcomes

- Concurrent GC by default.
- Reduce worst-case execution time.
- Preserve throughput.

Riptide

- Introduction
- Retreating Wavefront
- Space-Time Scheduler
- Safepoints
- Obstruction-Free Double Collect Snapshot
- Embedded WTF Locks
- Conclusion

Retreating Wavefront Write Barrier

- What is it?
- Why is it a good idea?
- What does everyone else do?

marked and scanned

marked but not scanned

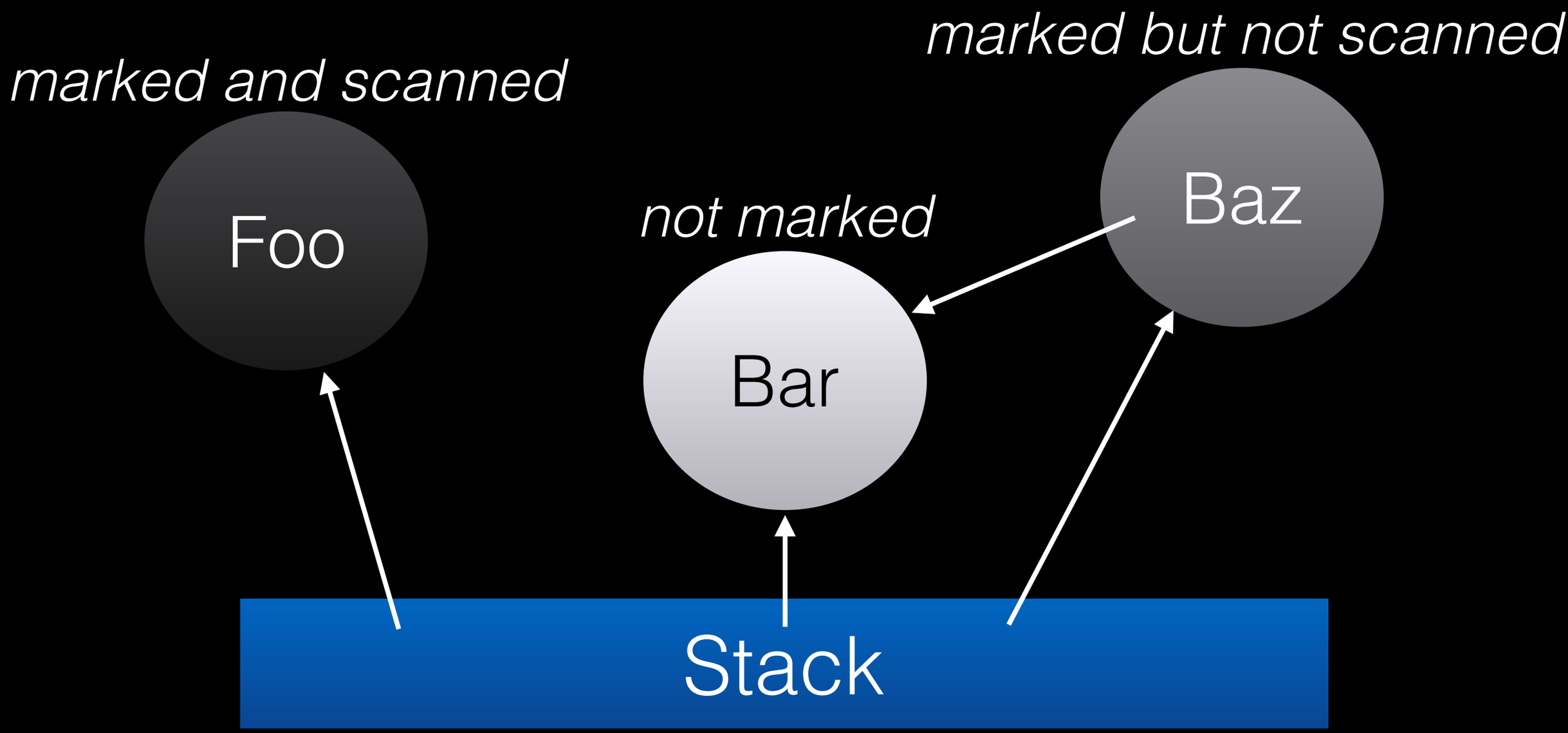
not marked

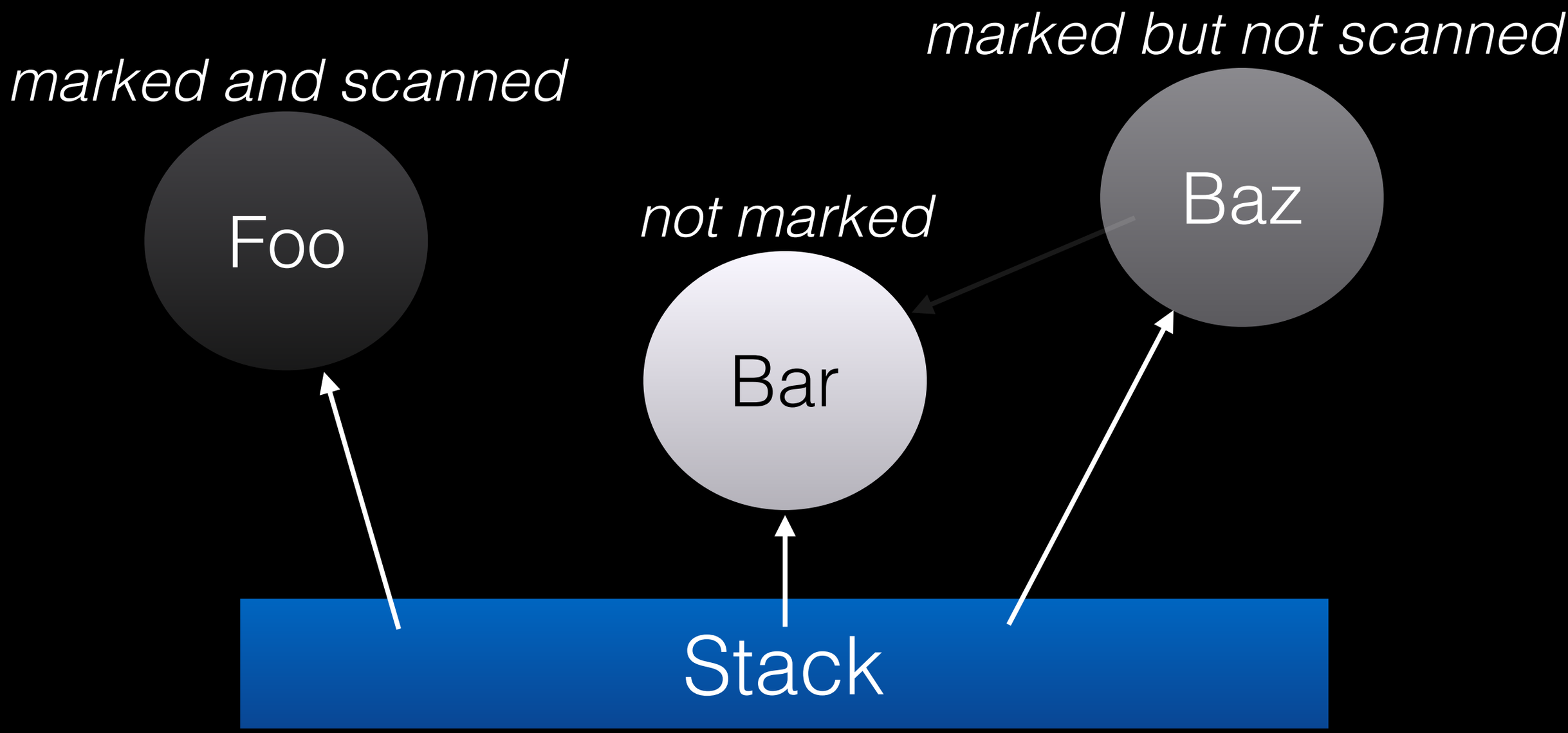
Foo

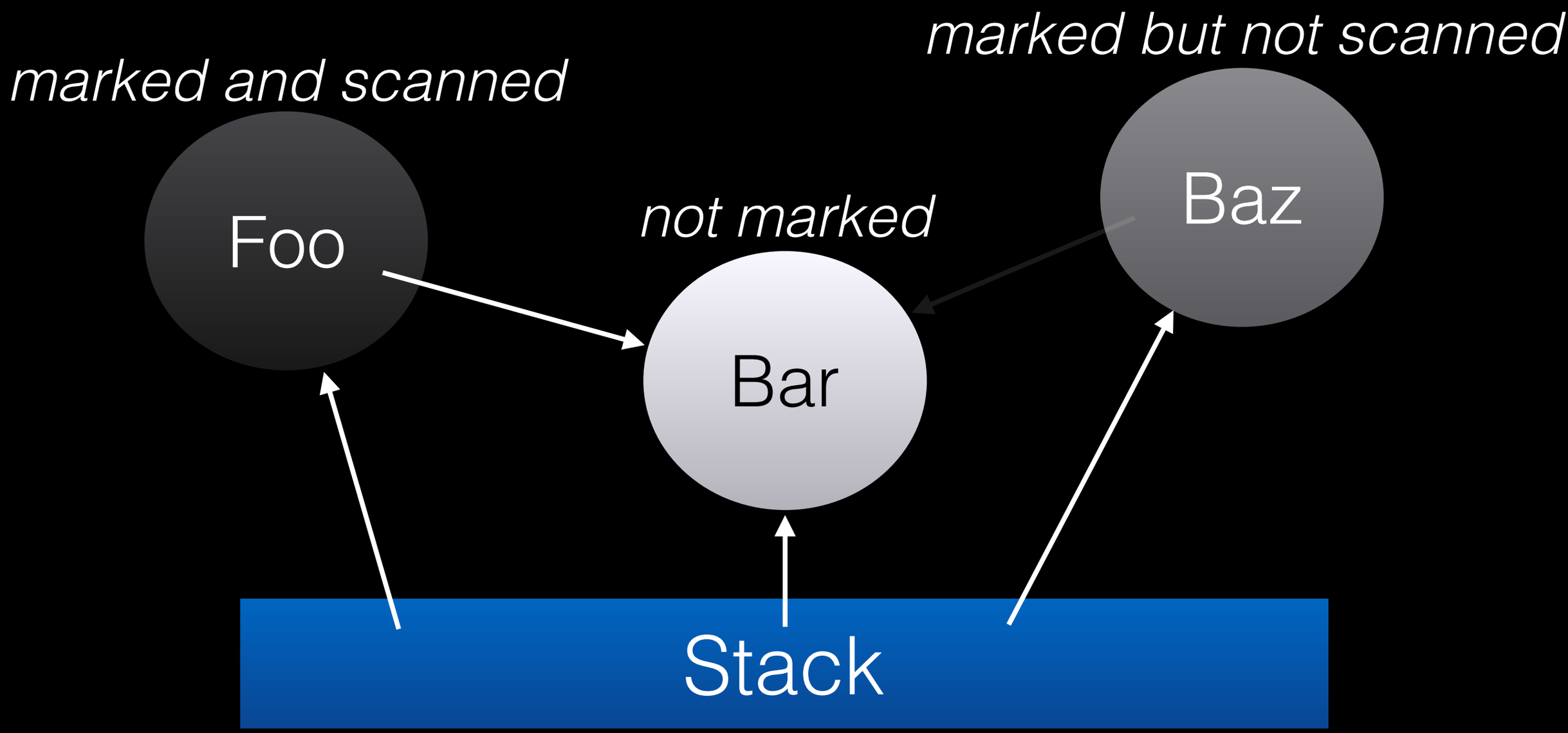
Bar

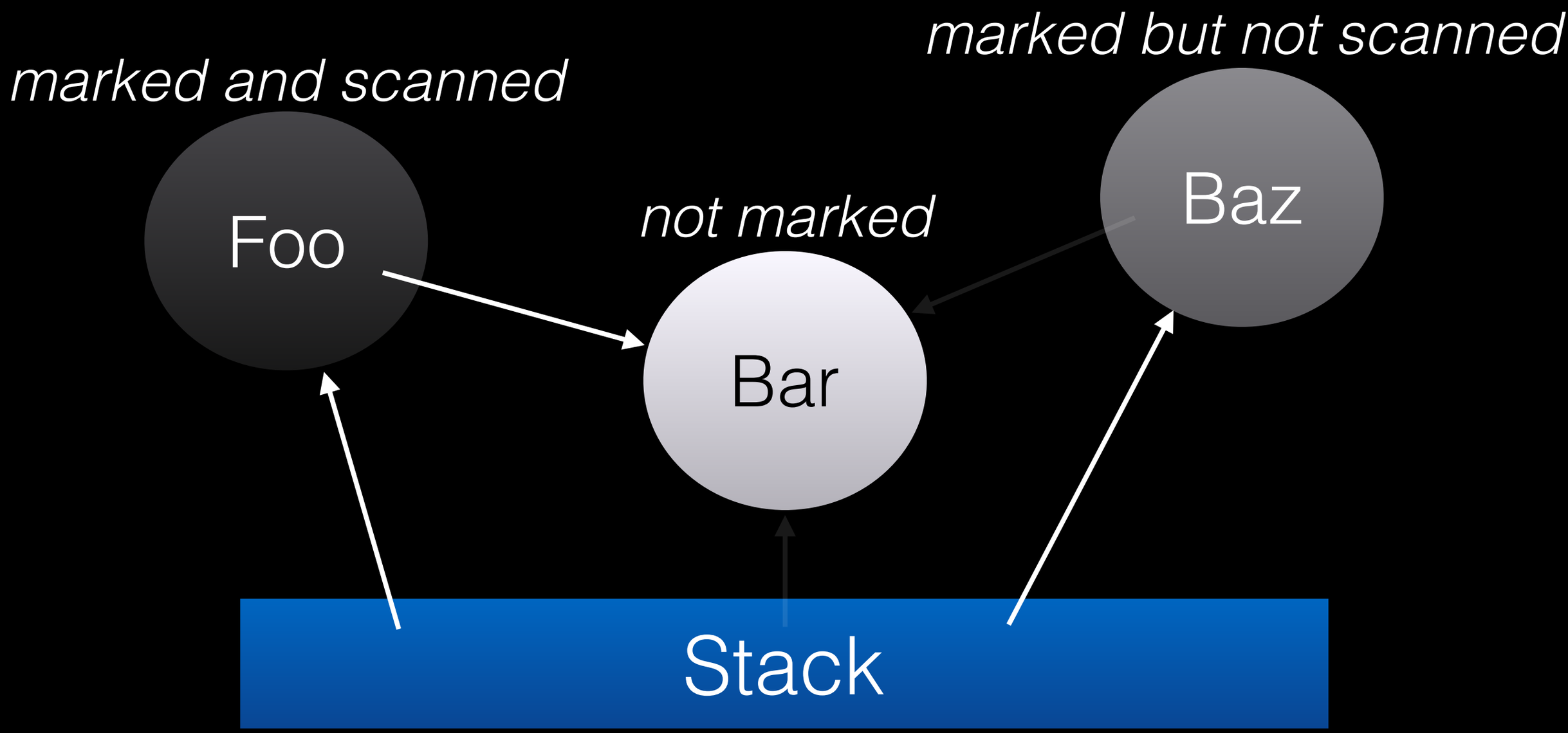
Baz

Stack


marked and scanned
marked but not scanned
Foo
not marked
Baz
Bar
Stack


marked and scanned
marked but not scanned
Foo
not marked
Baz
Bar
Stack


marked and scanned
marked but not scanned
Foo
not marked
Baz
Bar
Stack

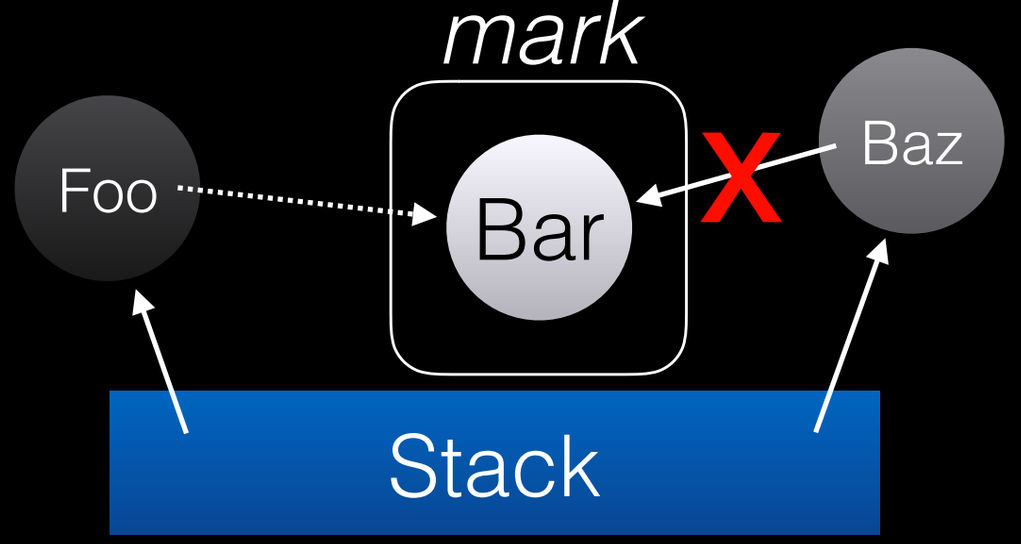

marked and scanned
marked but not scanned
Foo
not marked
Baz
Bar
Stack

Two major styles of barriers

- Advancing Wavefront
- Retreating Wavefront

Advancing Wavefront

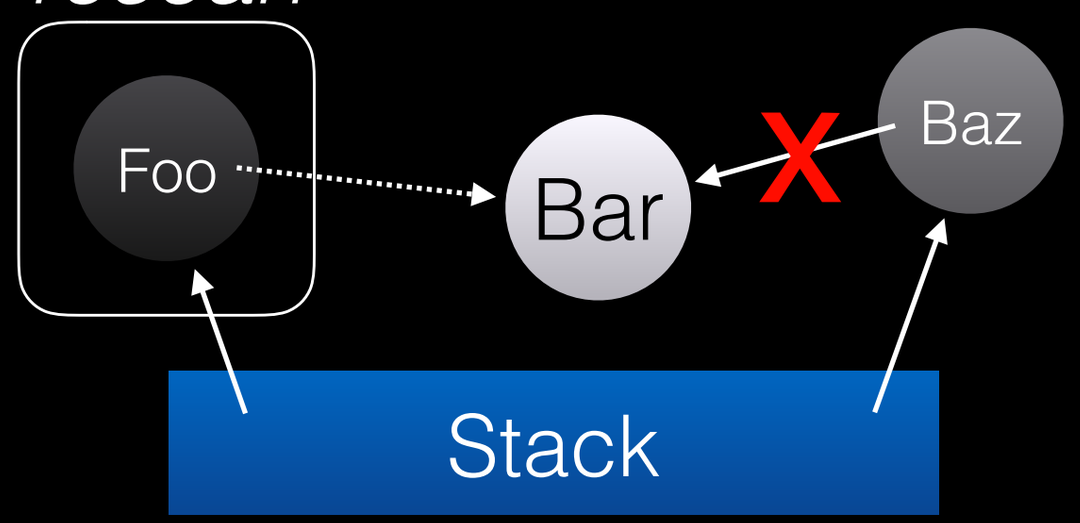
- *Mark objects involved in the race.*
- Dijkstra '76, Baker '78, Yuasa ’90.
- Super popular.



Retreating Wavefront

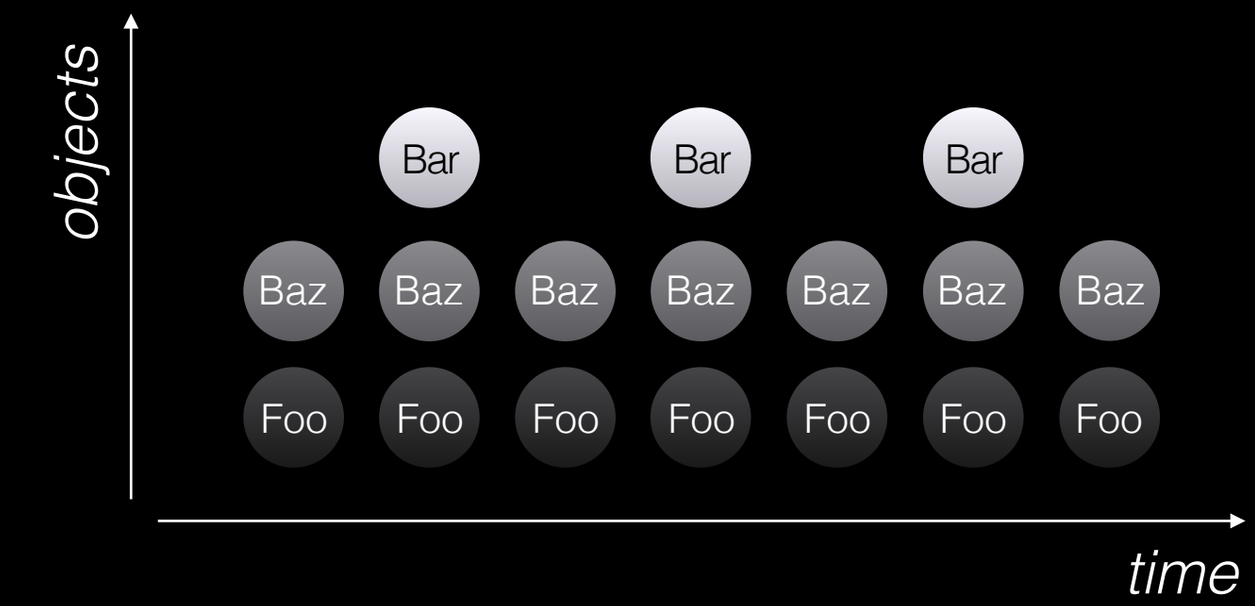
- *Rescan objects that got stored into.*

- Proposed by Guy Steele in 1975.

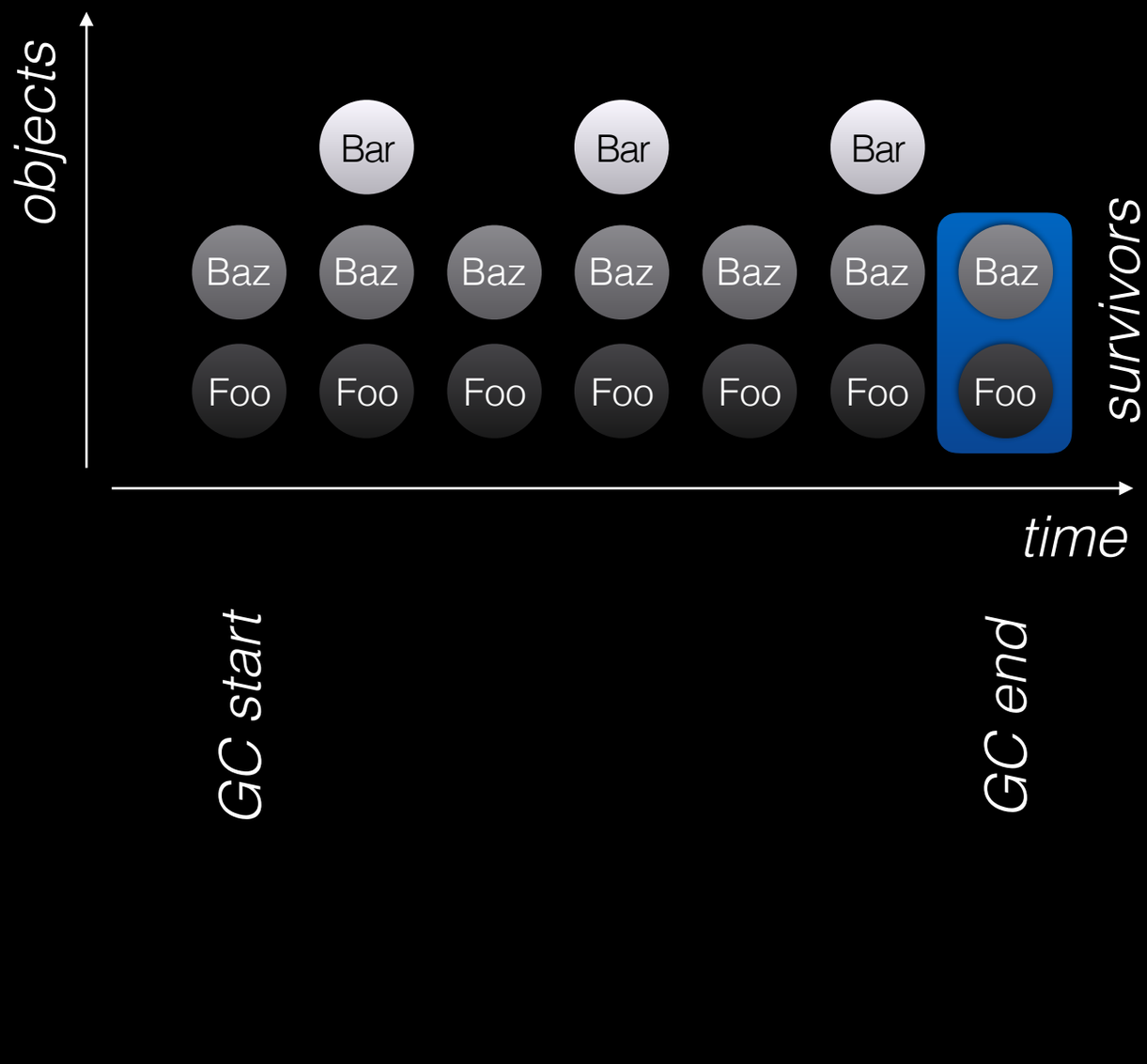


Style	Benefit
Advancing Wavefront	Guaranteed Forward Progress
Retreating Wavefront	Awesome Write Barrier

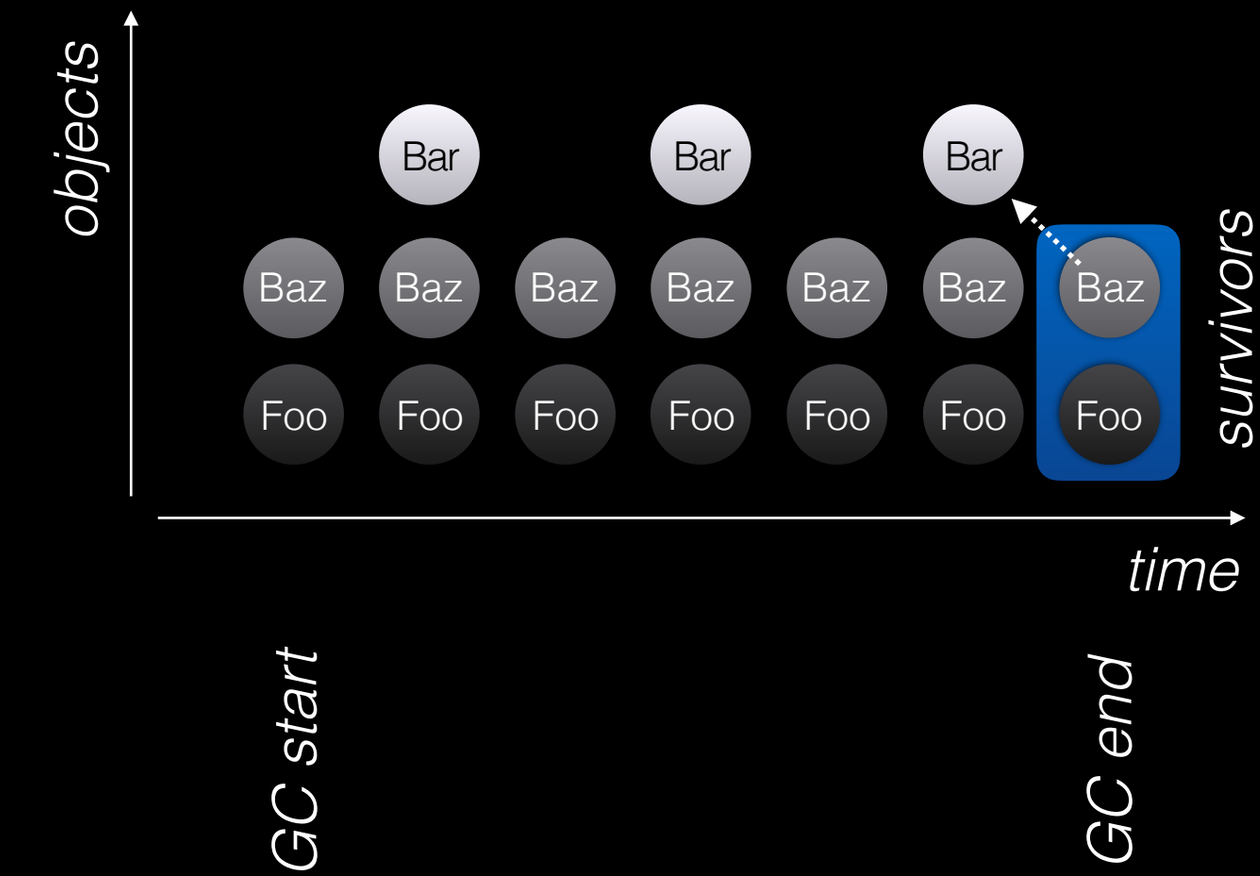
Retreating Wavefront = Delayed GC End



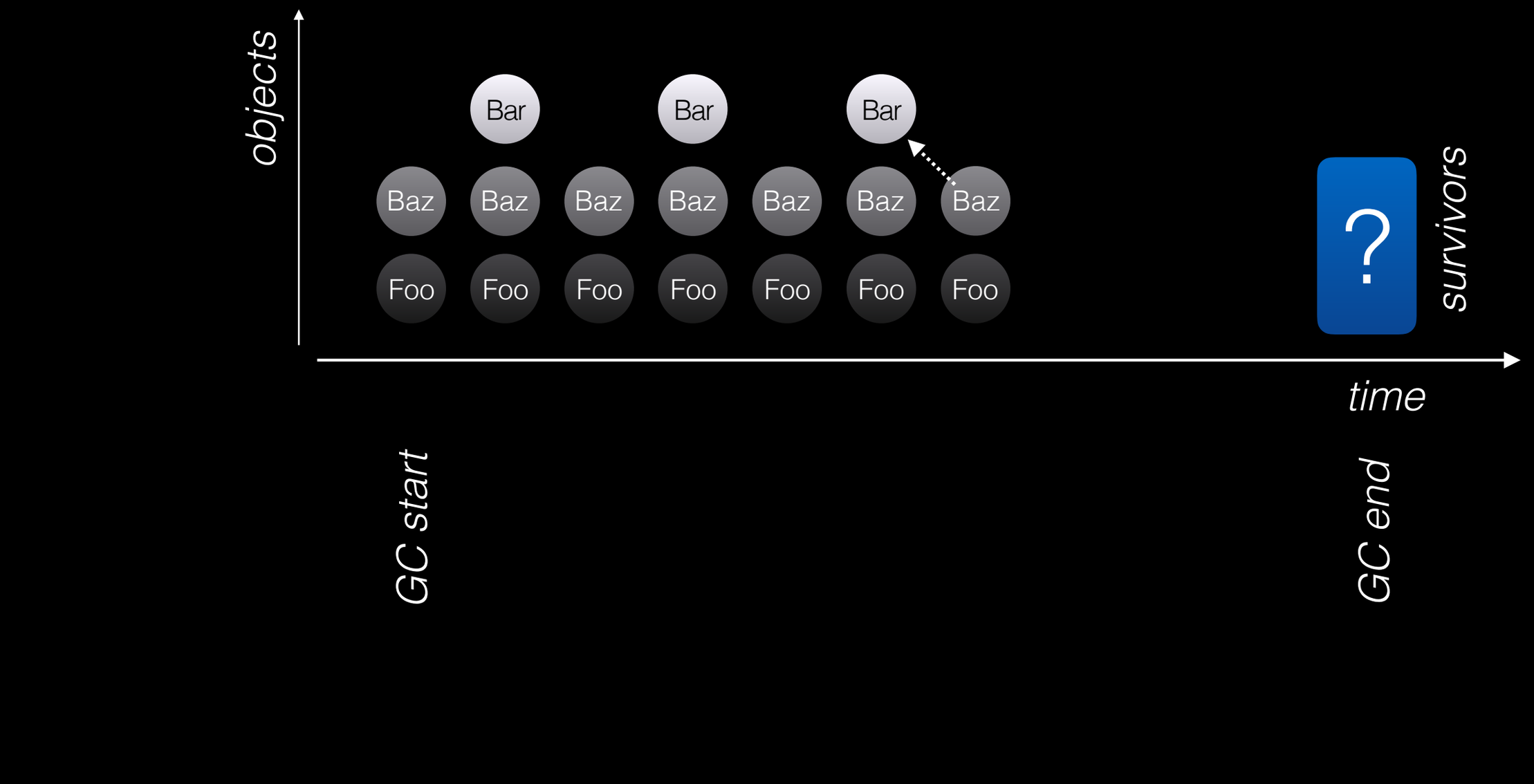
Retreating Wavefront = Delayed GC End



Retreating Wavefront = Delayed GC End



Retreating Wavefront = Delayed GC End



Old WebKit generational write barrier

```
targetObject->field = newValue;
if (newValue is an object
    && targetObject is old)
    slowPath(targetObject);
```

Retreating Wavefront write barrier

```
targetObject->field = newValue;
if (newValue is an object
    && targetObject is black)
    slowPath(targetObject);
```

Retreating Wavefront

- Awesome Write Barrier
- Progress Not Guaranteed

Retreating Wavefront

- Awesome Write Barrier
- Progress Not Guaranteed

Death Spiral

Death Spiral

- GC cannot keep up with allocation.
- Program runs out of memory before GC finishes.

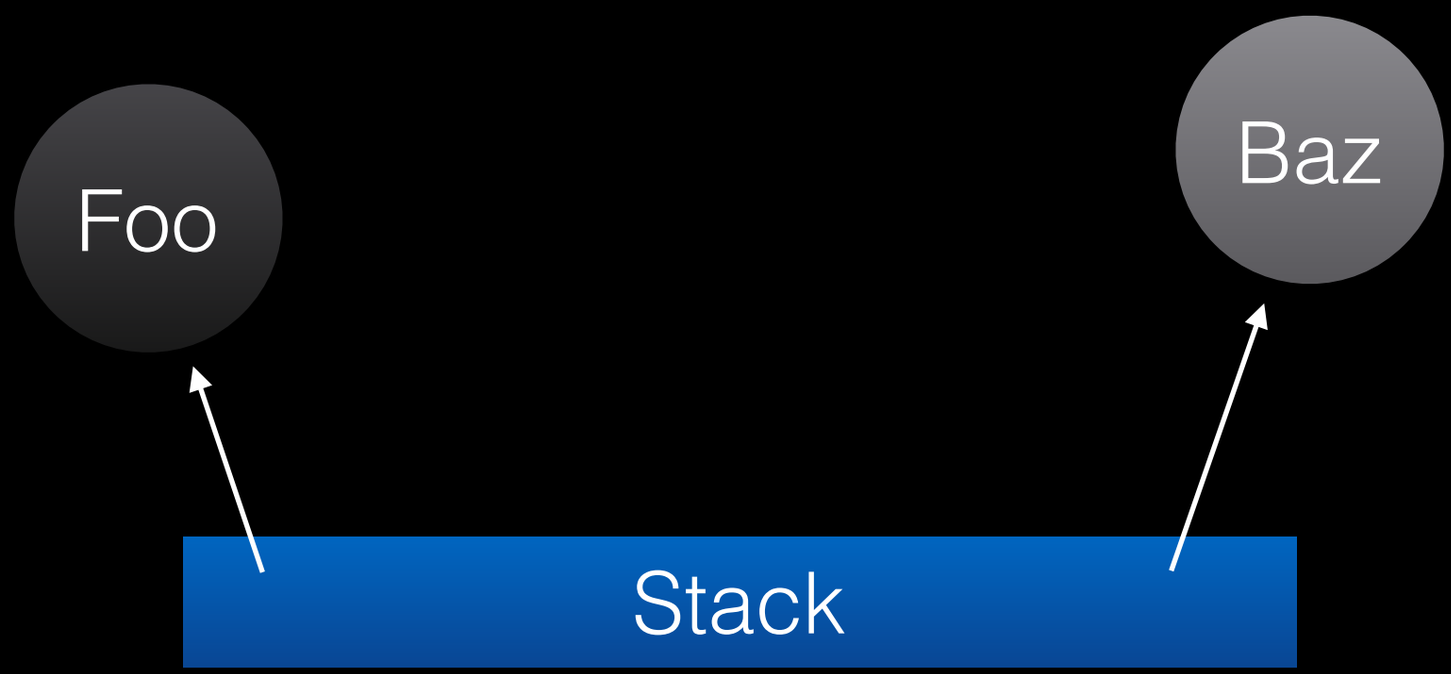
Style	Death Spiral
Advancing Wavefront	
Retreating Wavefront	Delayed GC End

Style	Death Spiral
Advancing Wavefront	Floating Garbage
Retreating Wavefront	Delayed GC End

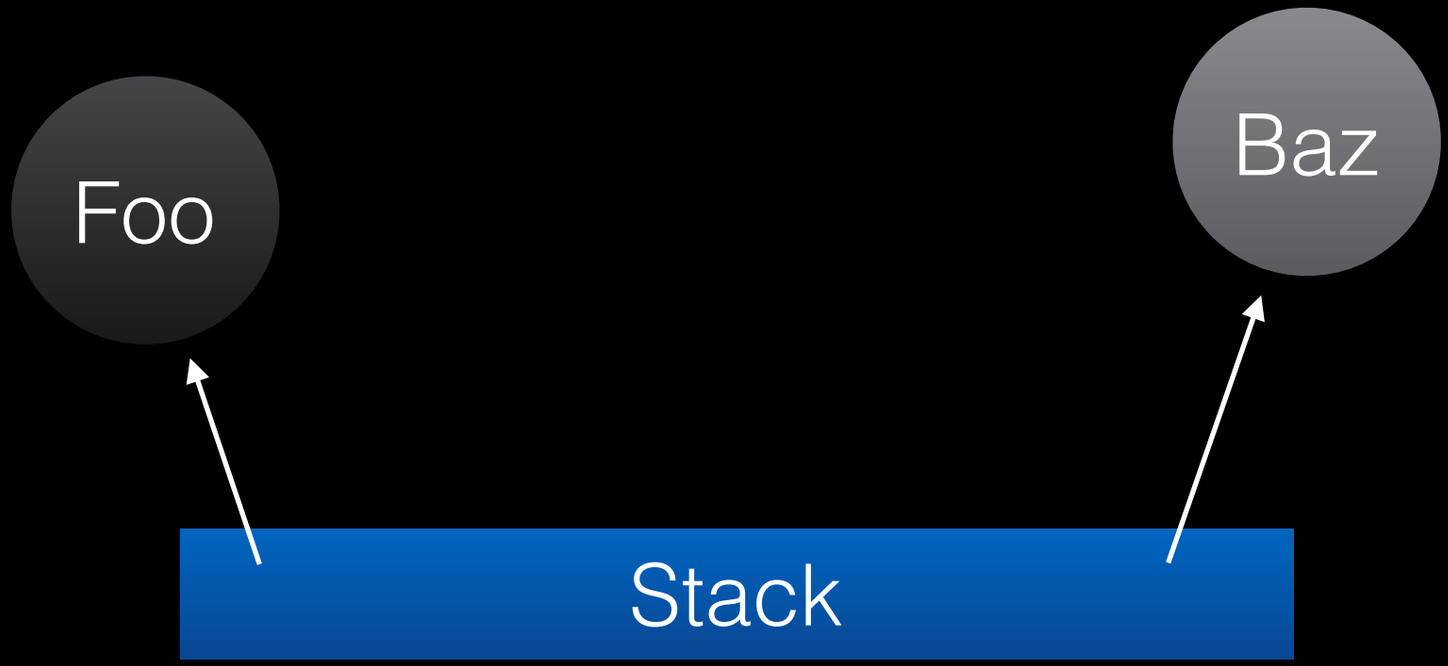
Advancing Wavefront = Floating Garbage

Retreating Wavefront

Advancing Wavefront



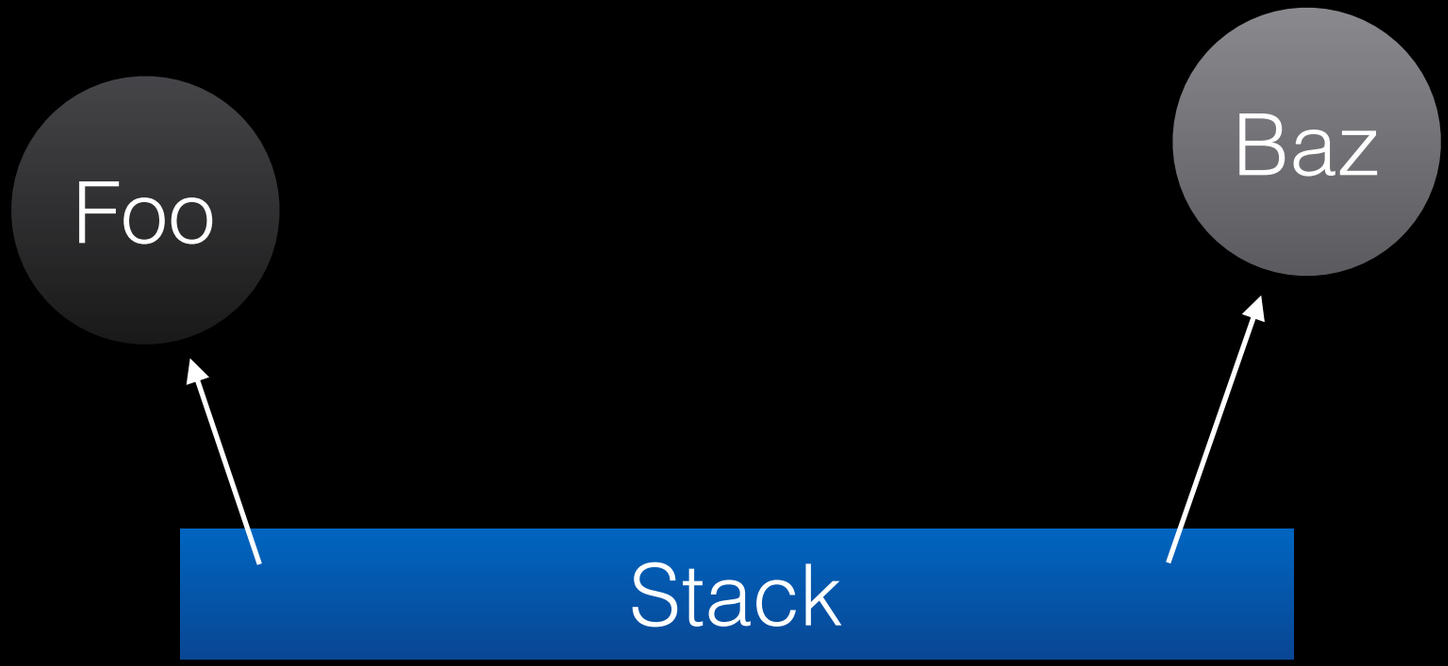
Advancing Wavefront = Floating Garbage



Advancing Wavefront = Floating Garbage

can delete bar1…bar5

must keep bar1…bar5

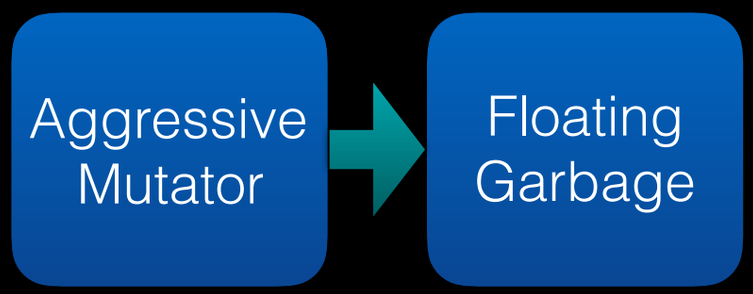


Advancing Wavefront Death Spiral

Advancing Wavefront Death Spiral

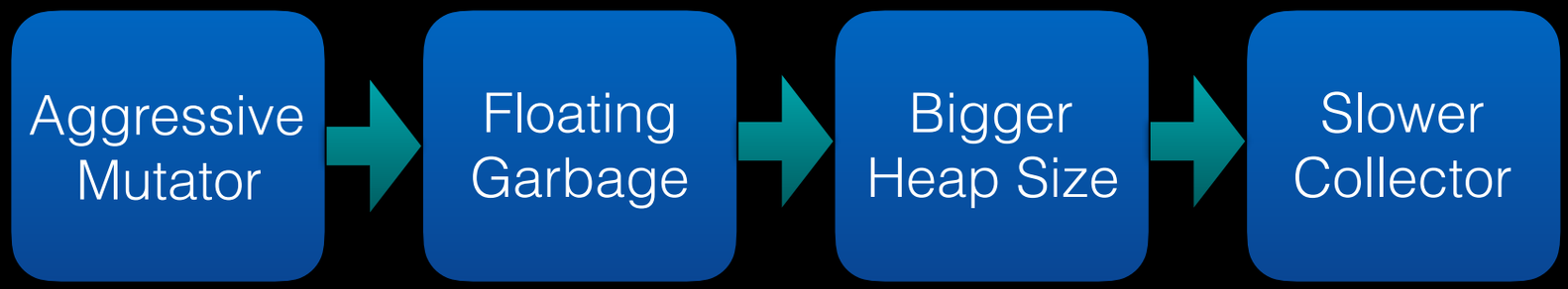
Aggressive Mutator

Advancing Wavefront Death Spiral



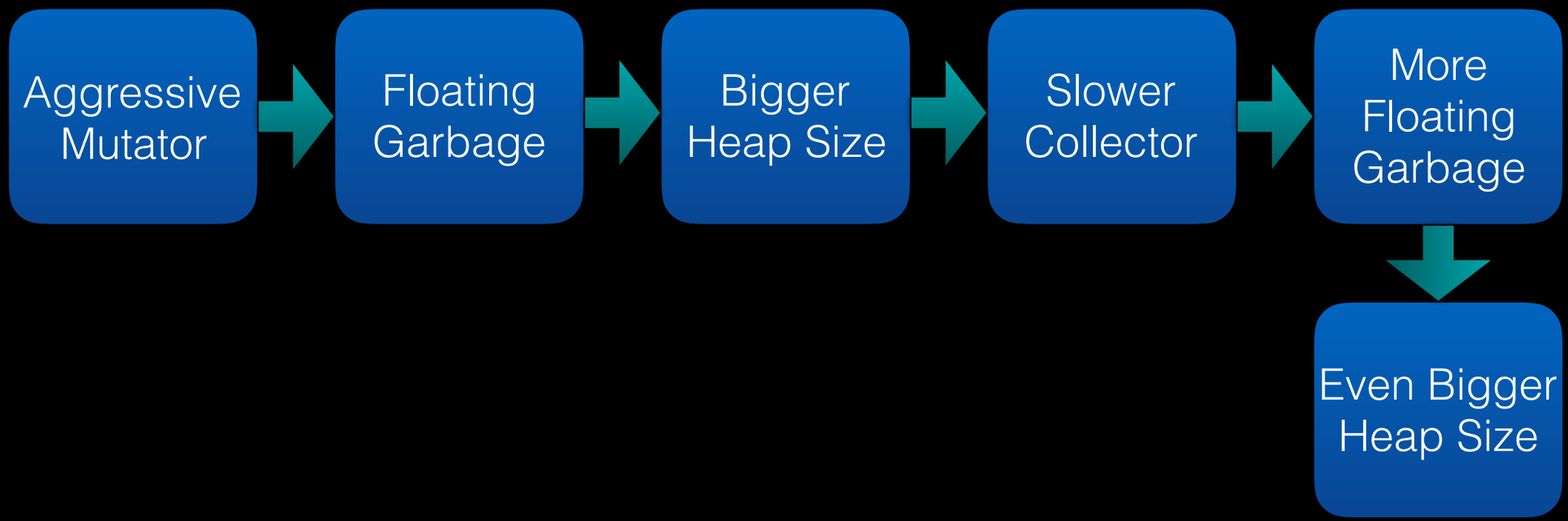
Advancing Wavefront Death Spiral

Advancing Wavefront Death Spiral

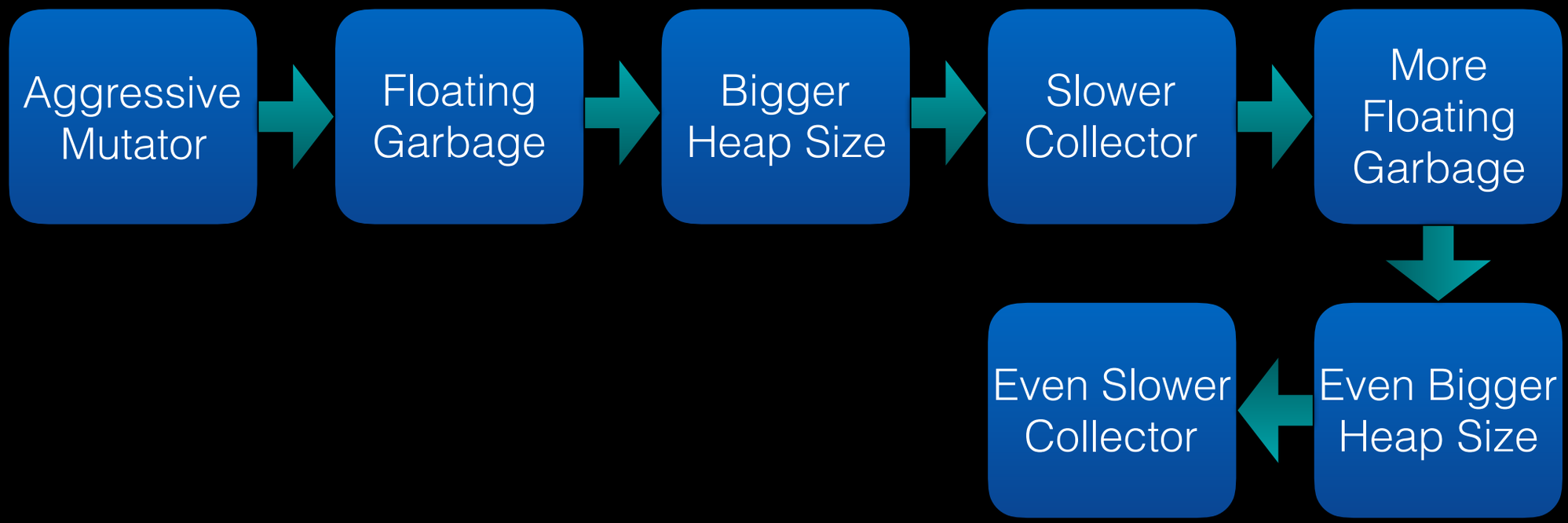


Advancing Wavefront Death Spiral

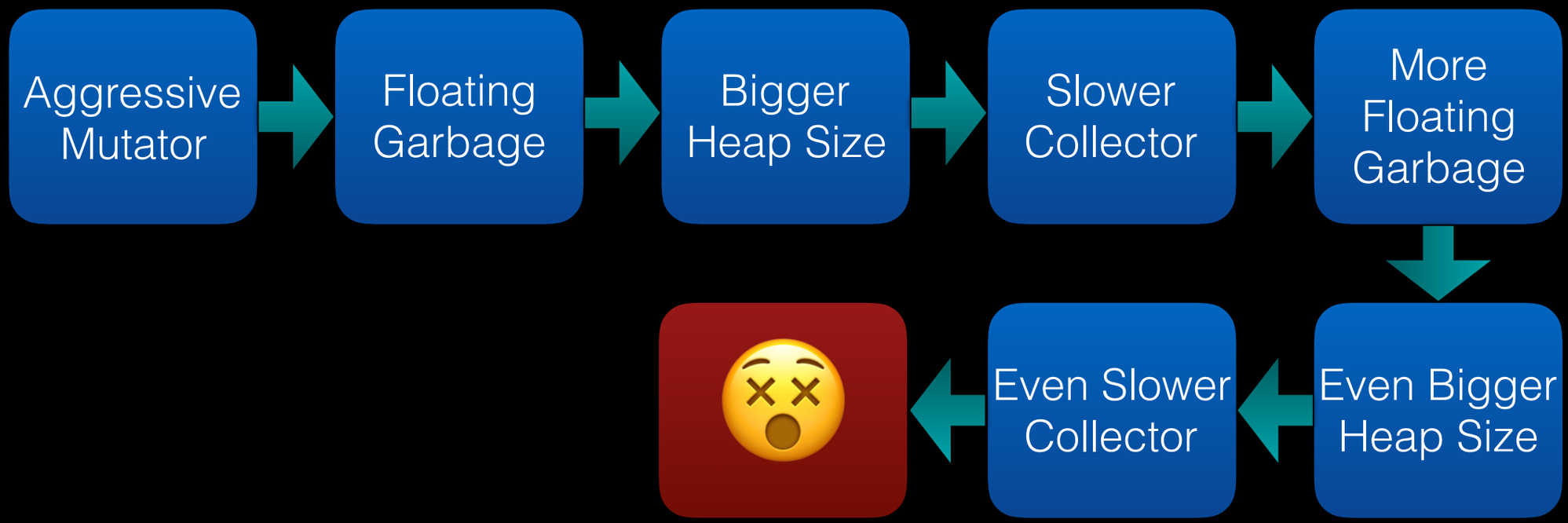
Advancing Wavefront Death Spiral



Advancing Wavefront Death Spiral



Advancing Wavefront Death Spiral



Style	Death Spiral	Mitigation
Advancing Wavefront	Floating Garbage	
Retreating Wavefront	Delayed GC End	

Style	Death Spiral	Mitigation
Advancing Wavefront	Floating Garbage	Fancy Scheduling
Retreating Wavefront	Delayed GC End	

Style	Death Spiral	Mitigation
Advancing Wavefront	Floating Garbage	Fancy Scheduling
Retreating Wavefront	Delayed GC End	?

Style	Death Spiral	Mitigation
Advancing Wavefront	Floating Garbage	Fancy Scheduling
Retreating Wavefront	Delayed GC End	Fancy Scheduling

Retreating Wavefront

- Cheaper barrier
- More death spiral mitigations

Death Spiral Mitigations

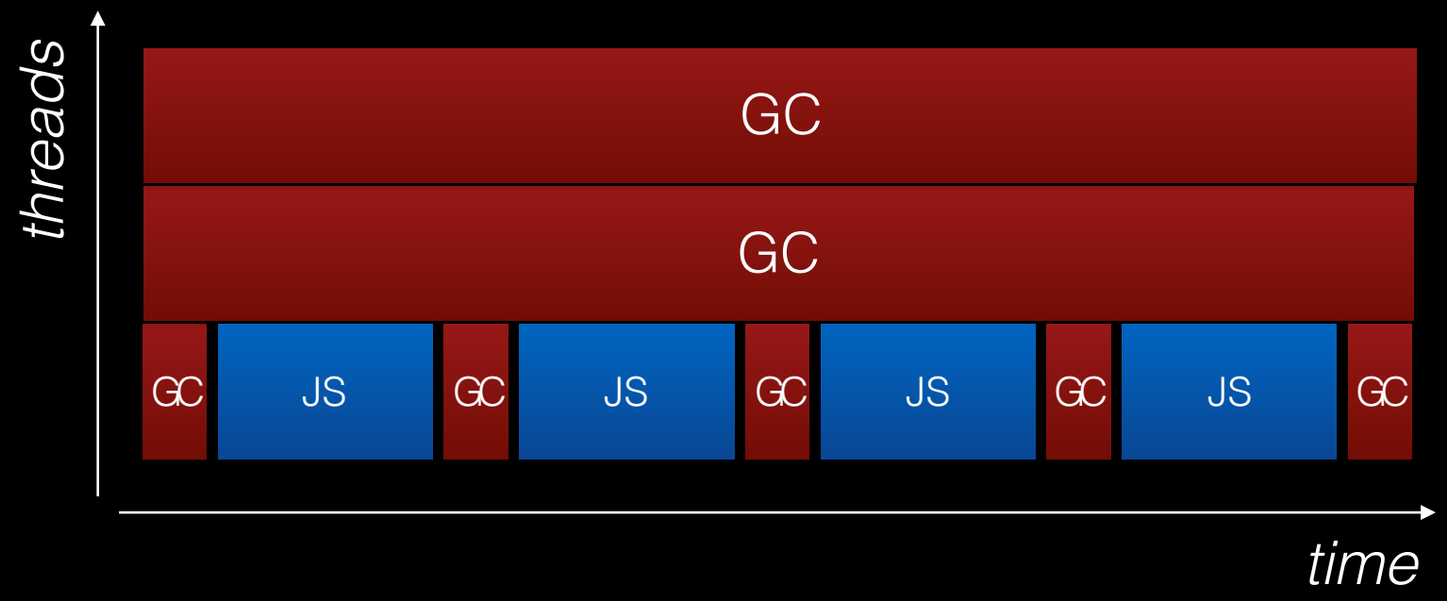
- Rescan requests are handled when collector runs out of all other work.

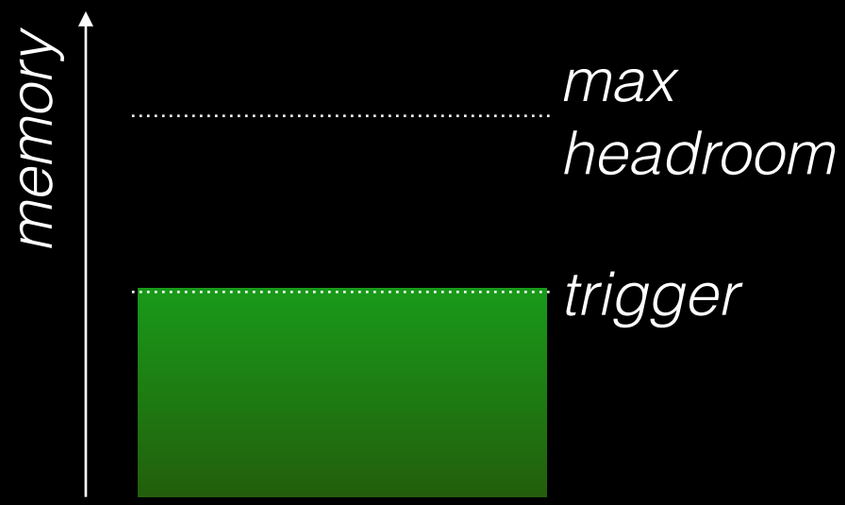
- Space-time Scheduler

Riptide

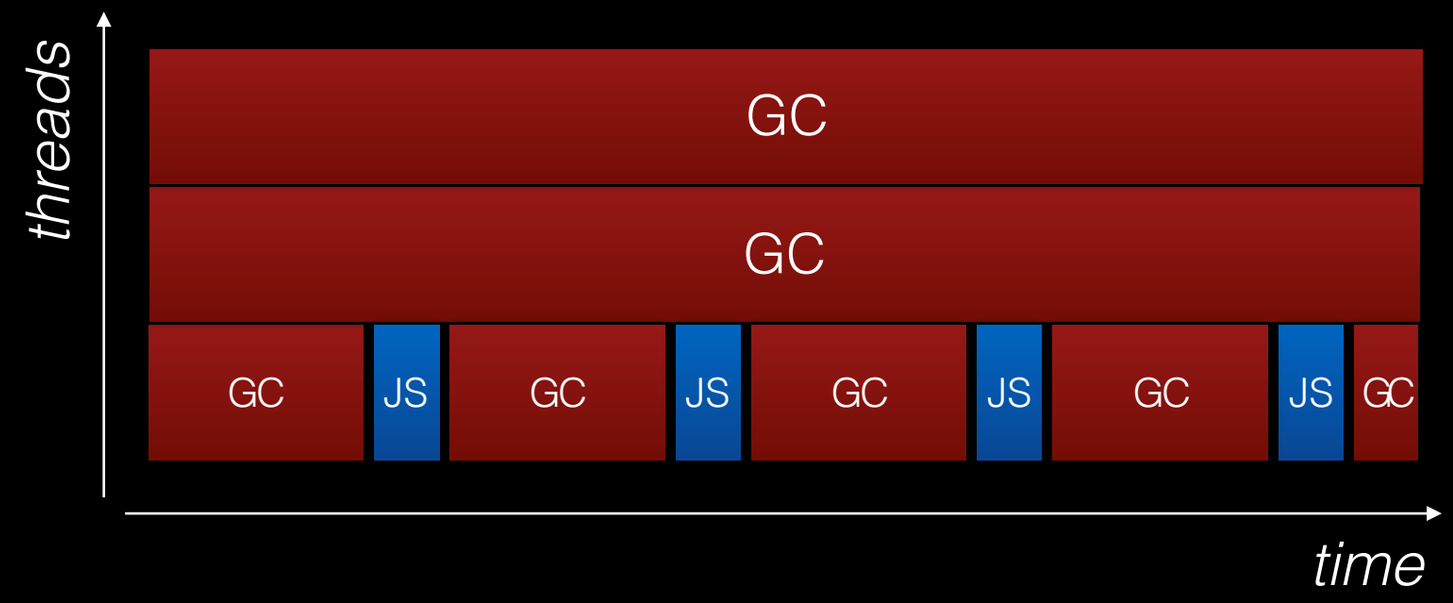
- Introduction
- Retreating Wavefront
- Space-Time Scheduler
- Safepoints
- Obstruction-Free Double Collect Snapshot
- Embedded WTF Locks
- Conclusion

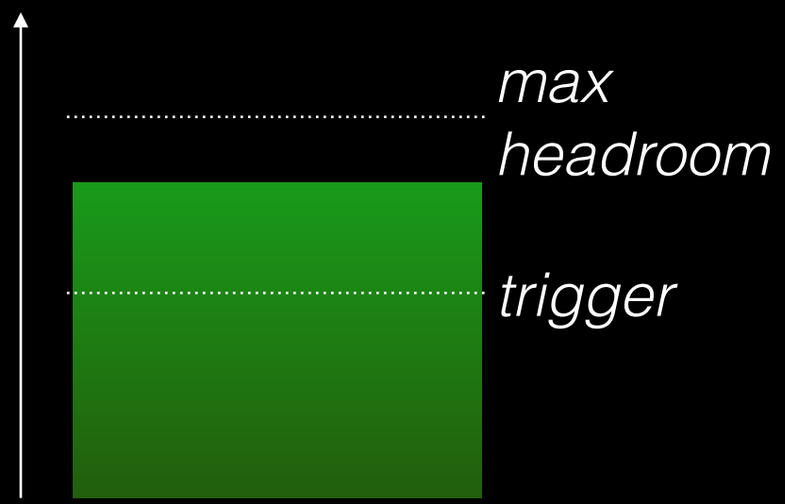
Space-Time Scheduler





Space-Time Scheduler





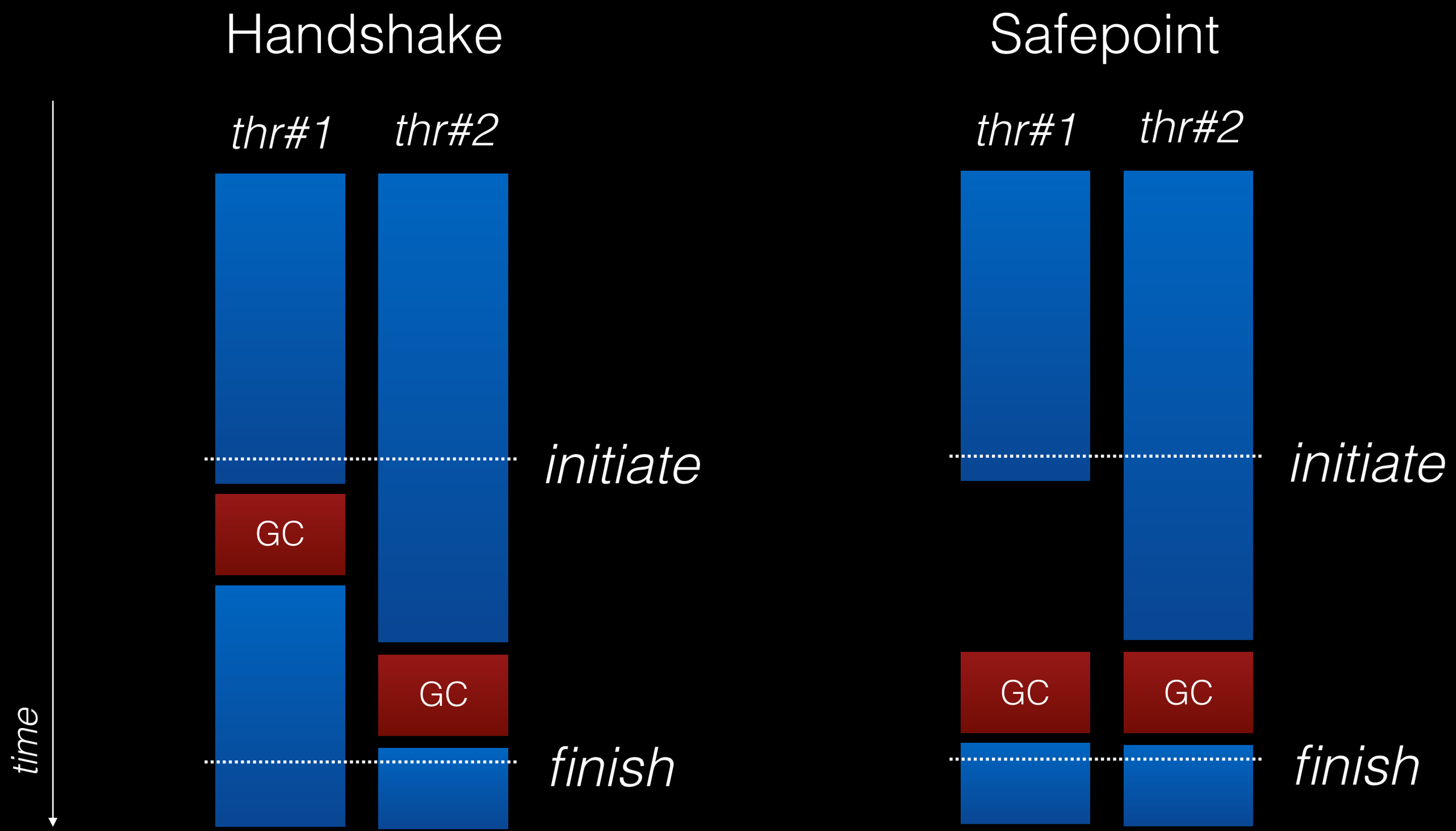
- The harder you push the collector, the harder it pushes back.

Riptide

- Introduction
- Retreating Wavefront
- Space-Time Scheduler
- Safepoints
- Obstruction-Free Double Collect Snapshot
- Embedded WTF Locks
- Conclusion

“Mostly Concurrent”

- All concurrent GCs have some work that needs to be performed on mutator threads.


Handshake
thr#1
thr#2
initiate
GC
GC
finish
time
Safepoint
thr#1
thr#2
initiate
GC
GC
finish

Style	Pro	Con
Handshake	Lower Latency	Does not work with retreating wavefront Does not work with marking constraints
Safepoint	Easy to Implement	Higher Latency (if you have tons of threads)

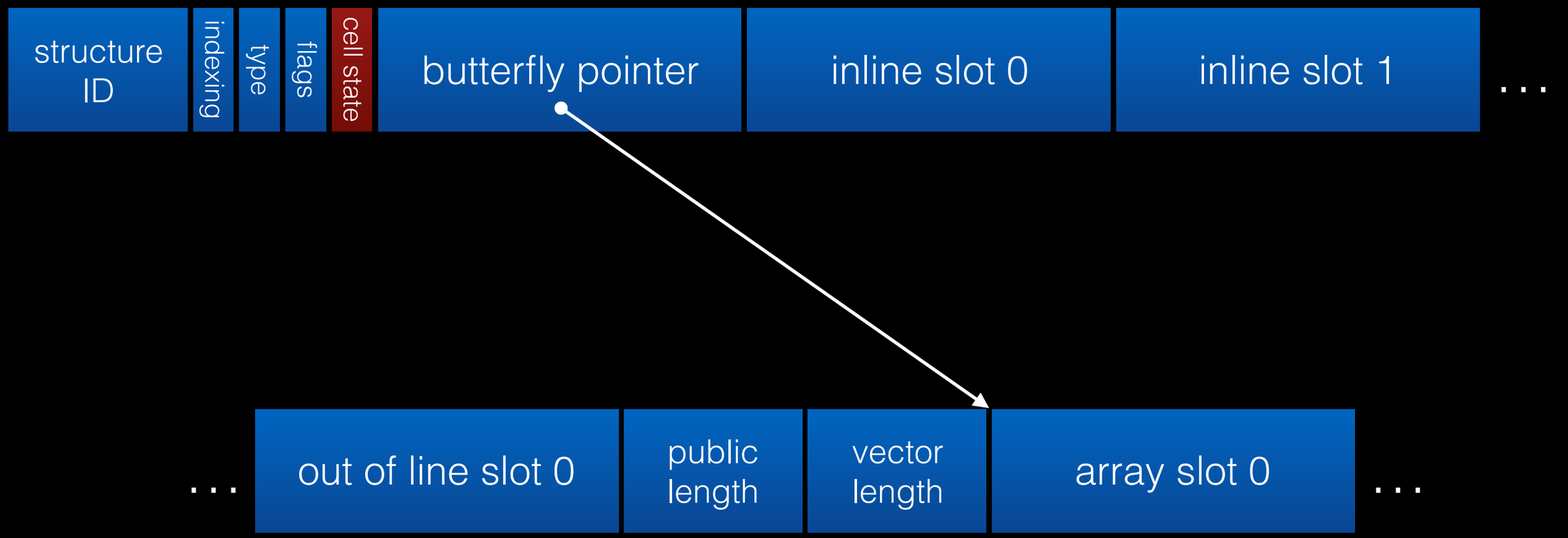
Things we do under safepoint

- Stack scanning
- Logically clearing mark bits
- Marking constraints
- Misc bookkeeping

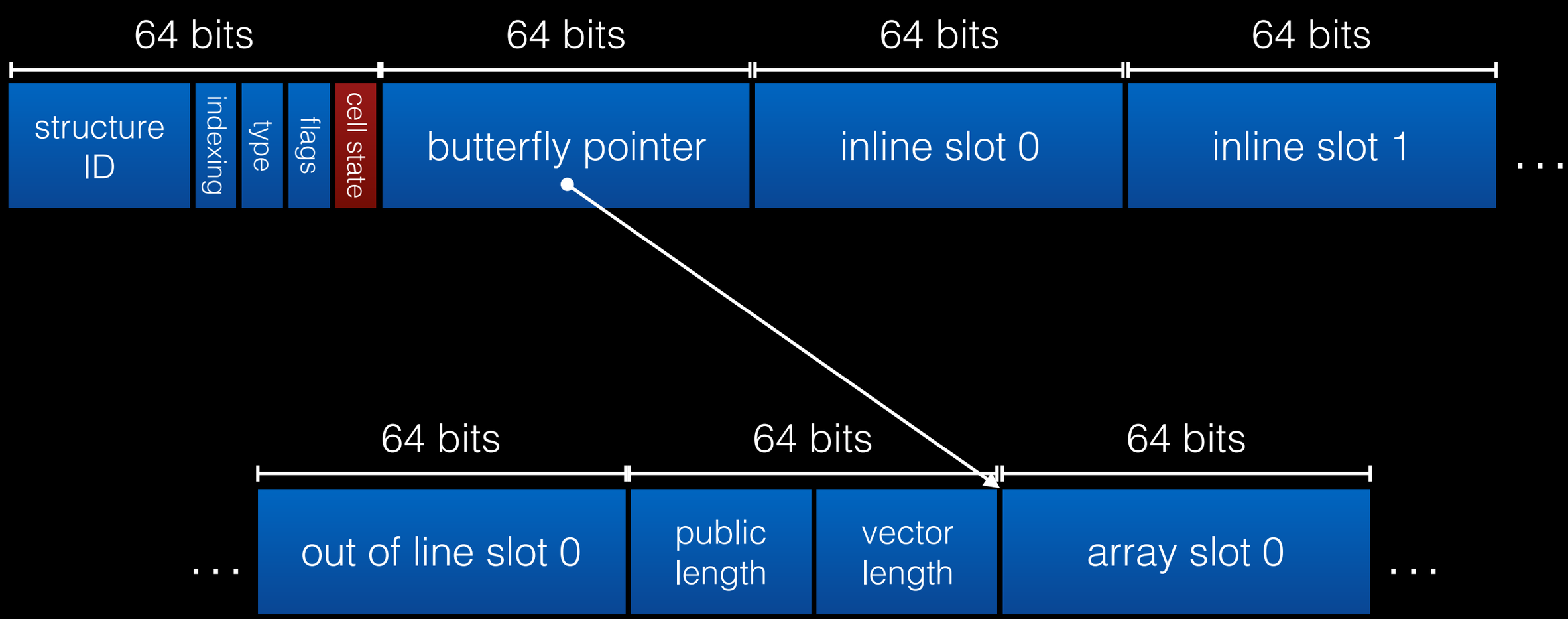
Riptide

- Introduction
- Retreating Wavefront
- Space-Time Scheduler
- Safepoints
- Obstruction-Free Double Collect Snapshot
- Embedded WTF Locks
- Conclusion

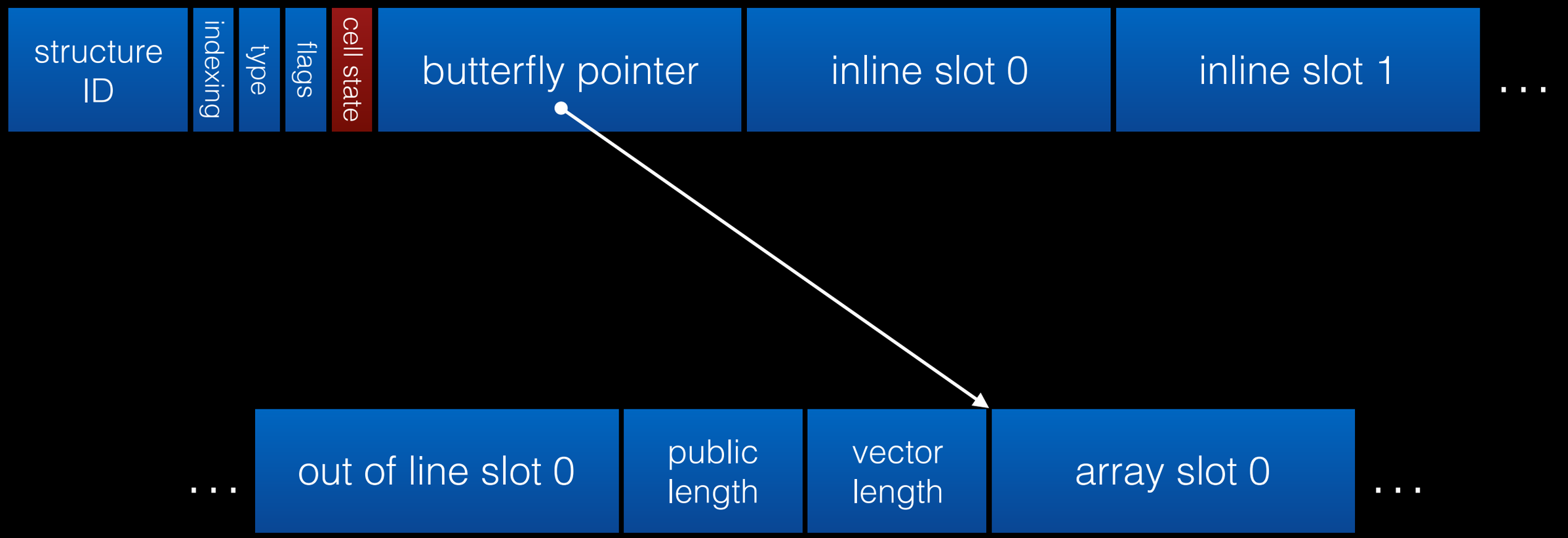
JSC Object Model



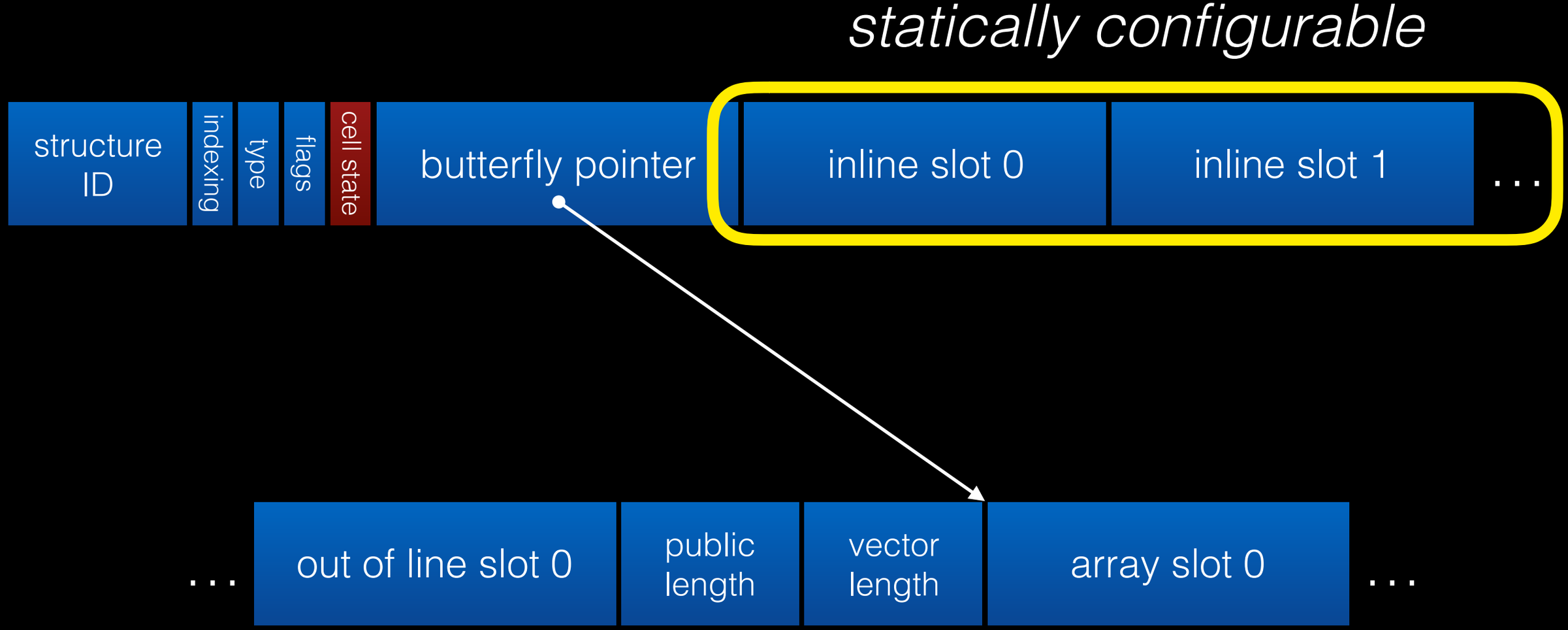
JSC Object Model



JSC Object Model

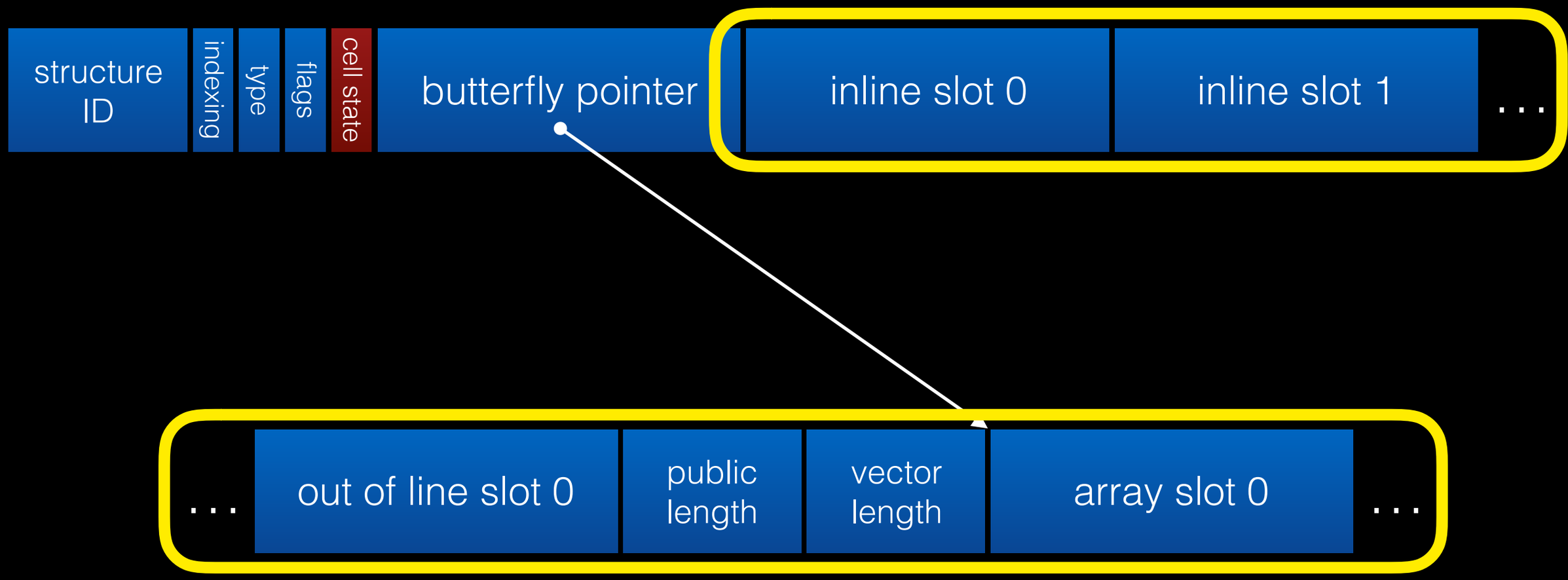


JSC Object Model



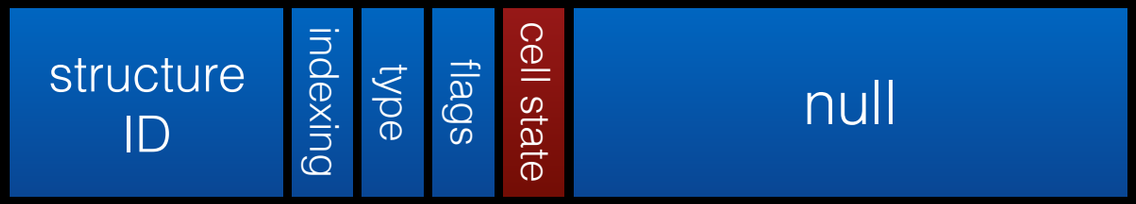
JSC Object Model

statically configurable



dynamically configurable

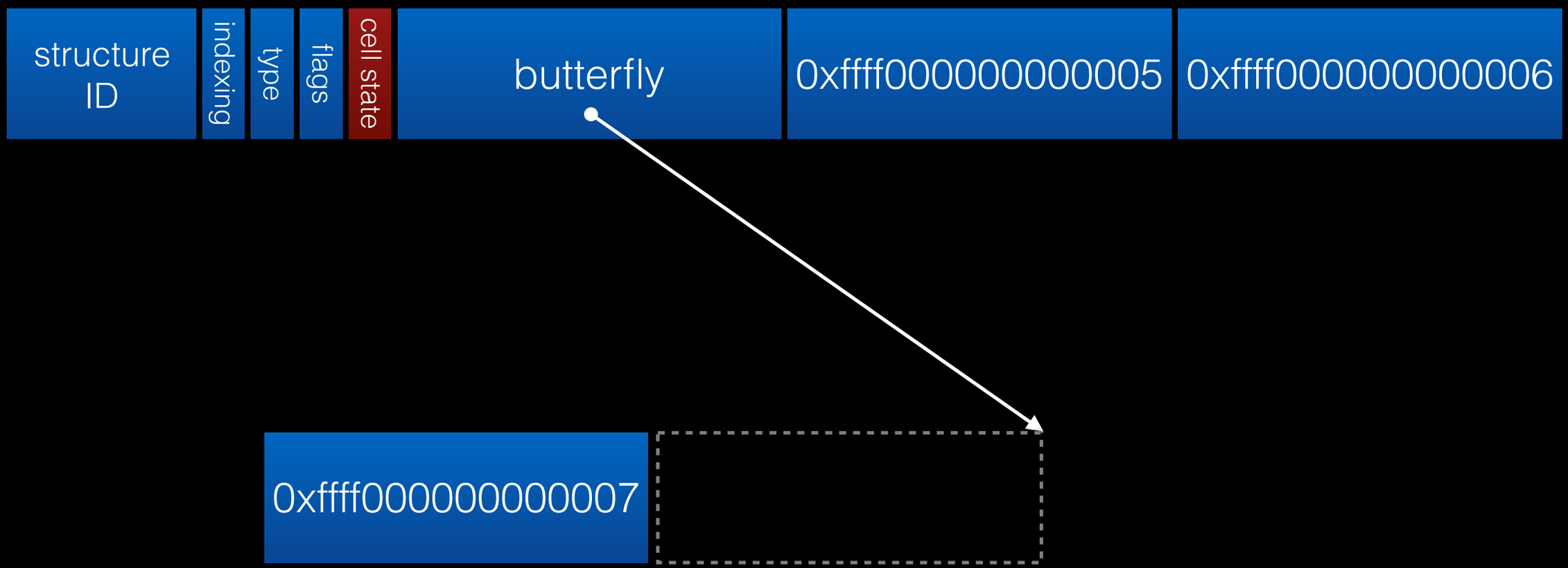
Empty JSObject



Fast JSObject

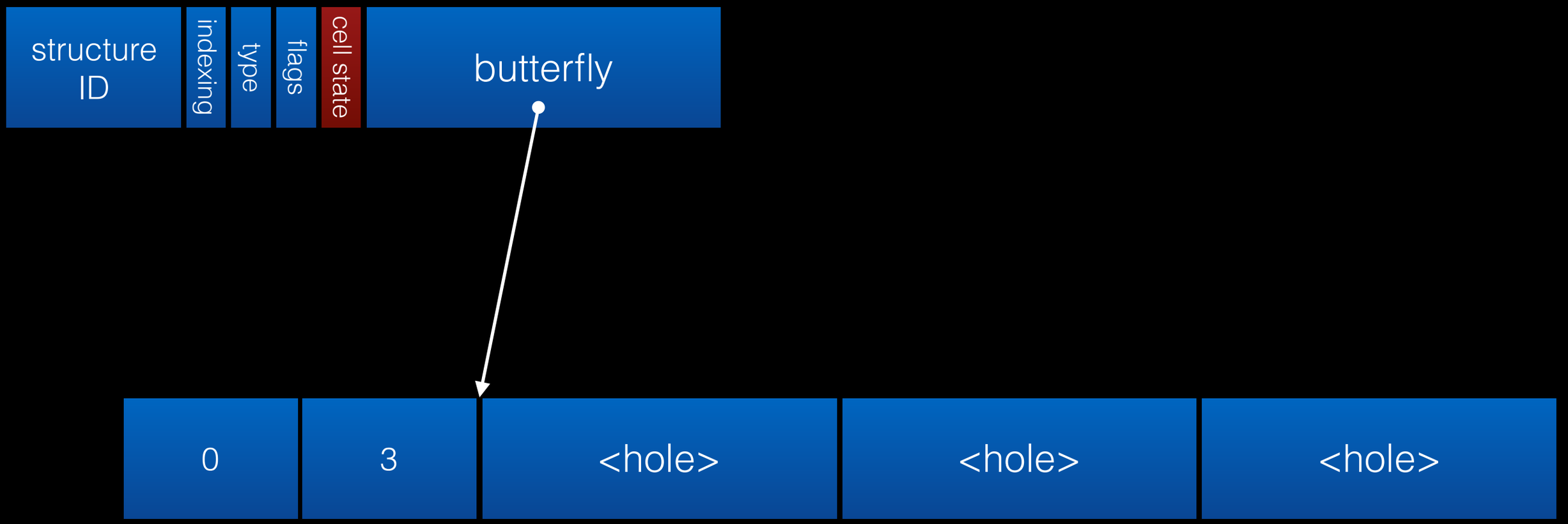
var o = {f: 5, g: 6};

JSObject with dynamically added fields



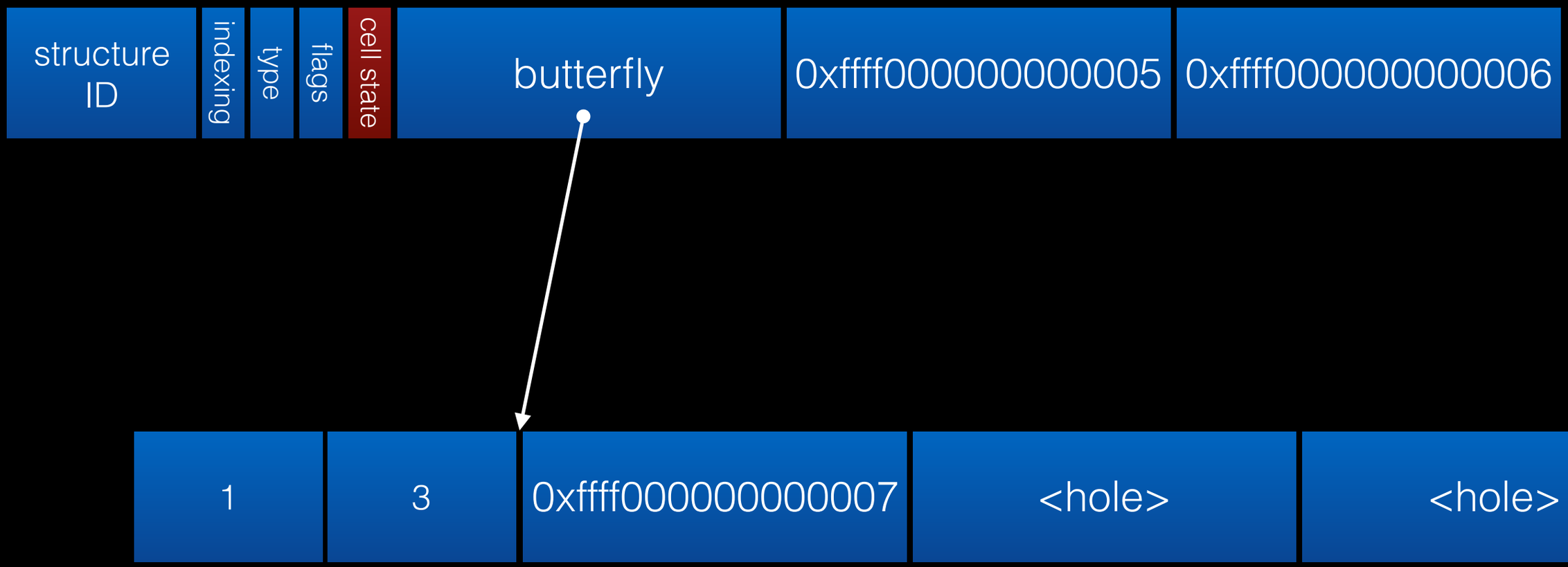

```
var o = {f: 5, g: 6};
o.h = 7;
```

JSArray with room for 3 array elements



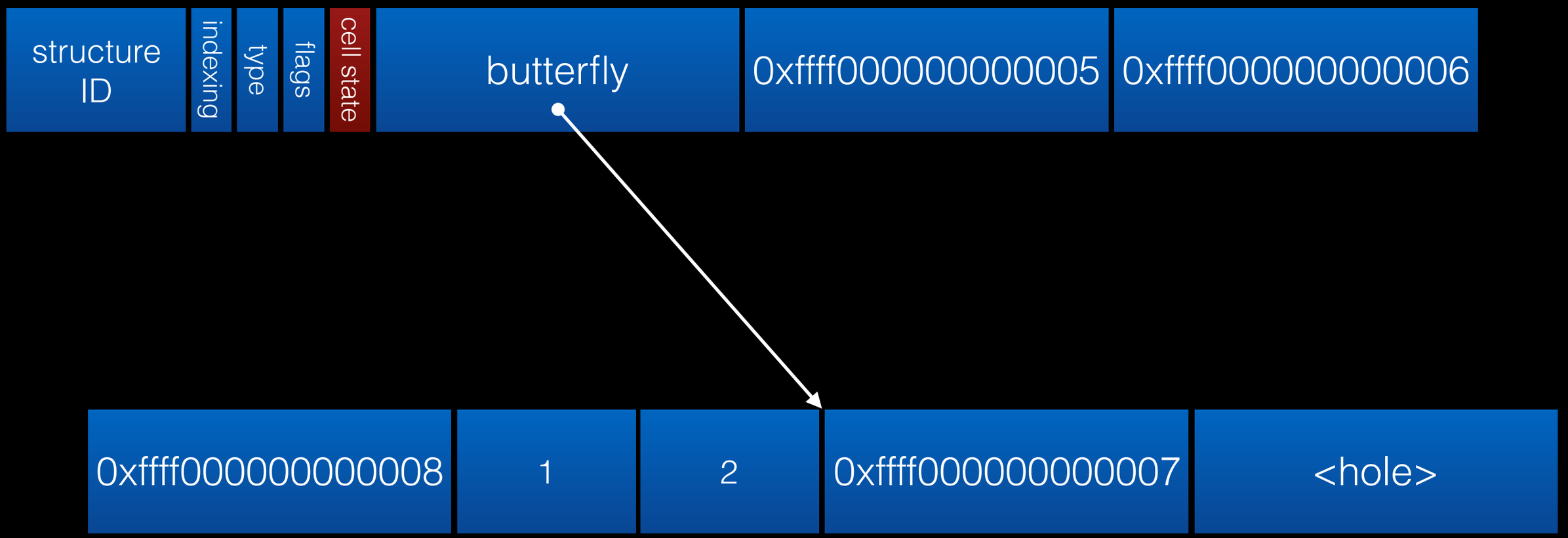
var a = [];

Object with fast properties and array elements



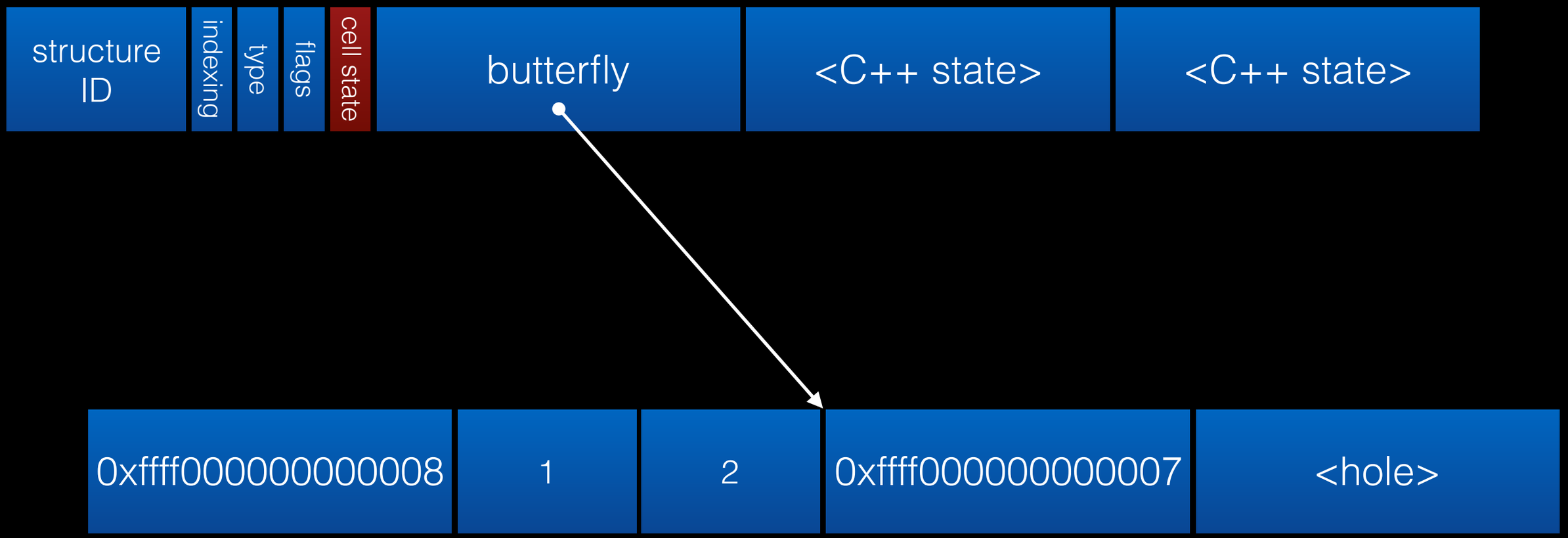

```
var o = {f: 5, g: 6};
o[0] = 7;
```

Object with fast and dynamic properties and array elements




```
var o = {f: 5, g: 6};
o[0] = 7;
o.h = 8;
```

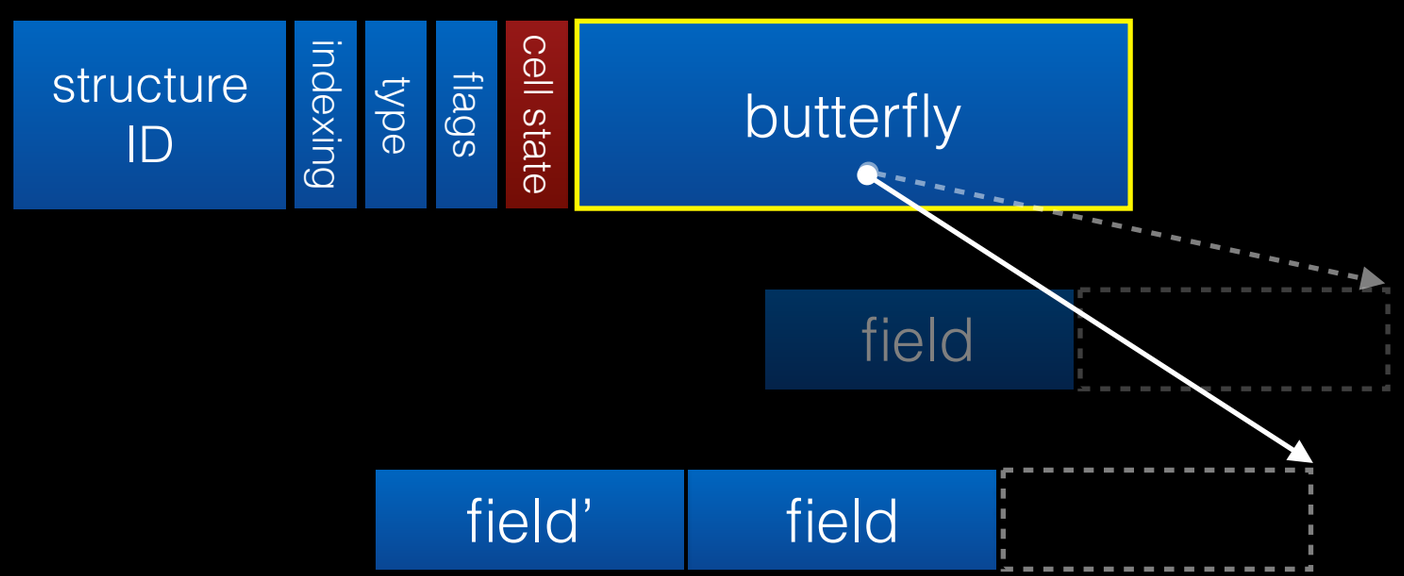
Exotic object with dynamic properties and array elements



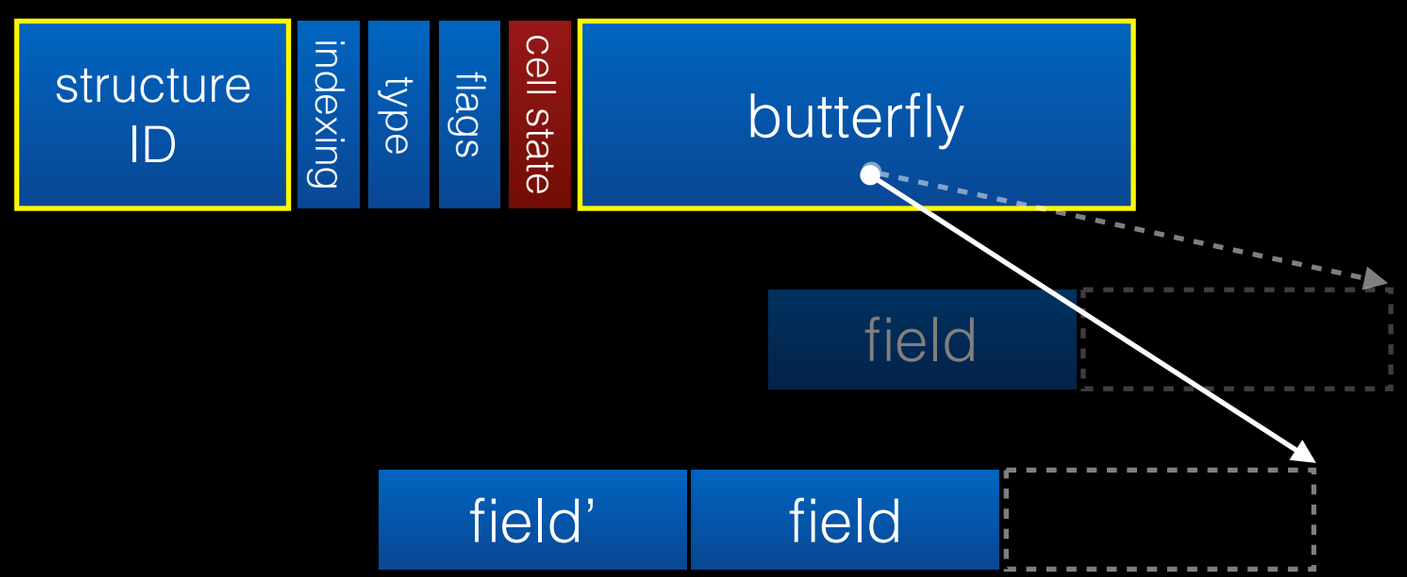

```
var o = new Date();
o[0] = 7;
o.h = 8;
```


structure ID
indexing
type
flags
cell state
butterfly
field

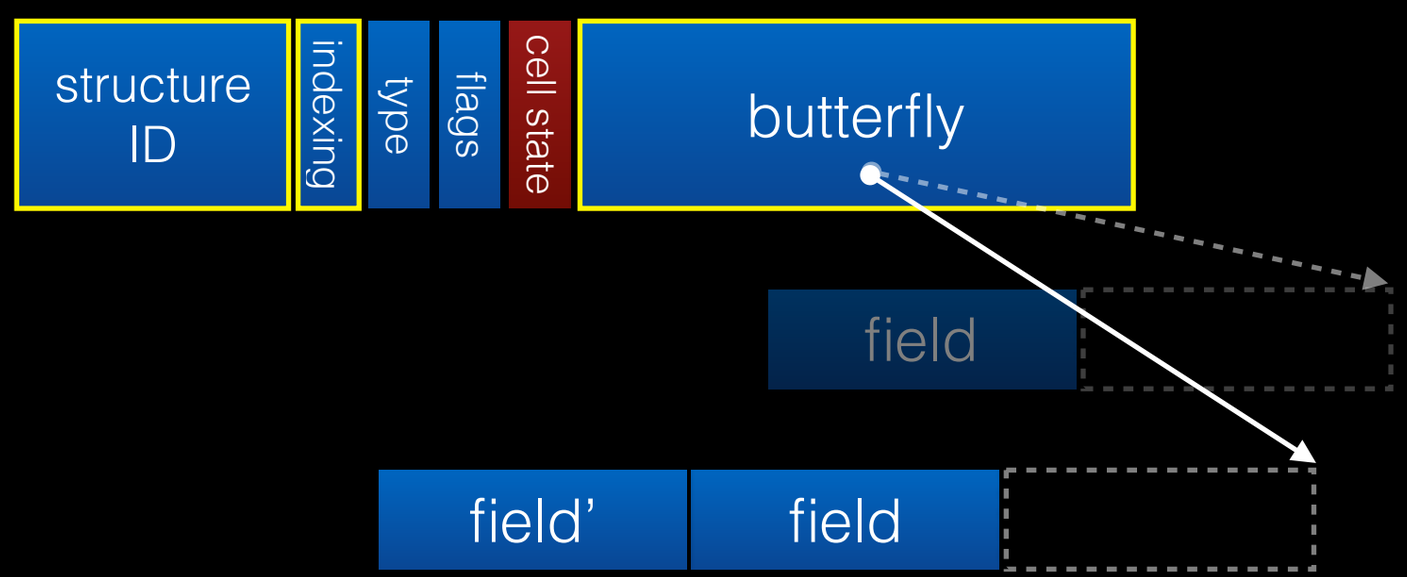
- Adding or removing properties (o.f, o[“f”]) and array elements (o[0], o[1], …) changes the butterfly.



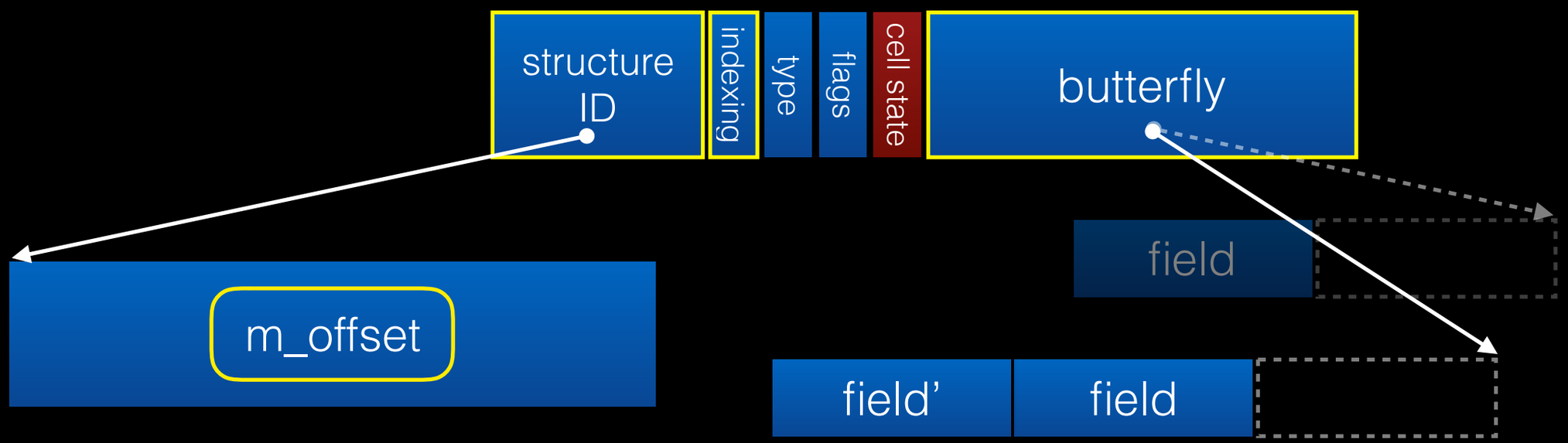
- Adding or removing properties (o.f, o[“f”]) and array elements (o[0], o[1], …) changes the butterfly.
- It also changes the structure ID.



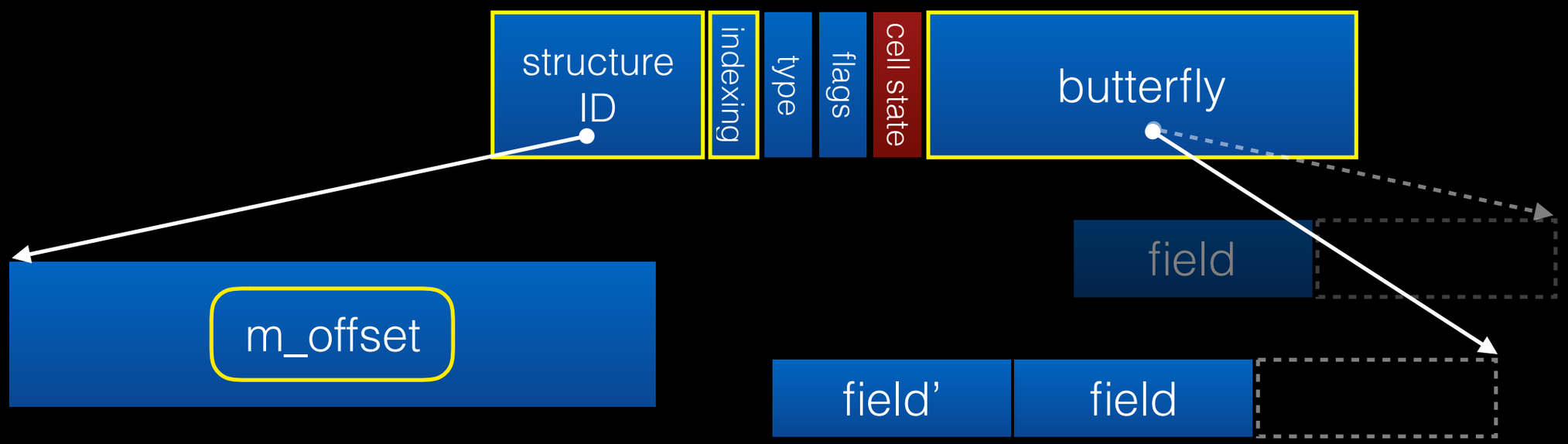
- Adding or removing properties (o.f, o[“f”]) and array elements (o[0], o[1], …) changes the butterfly.
- It also changes the structure ID.
- And possibly the indexing type.



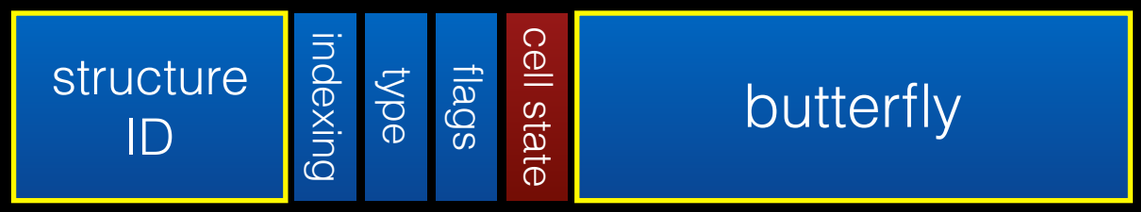
- Adding or removing properties (o.f, o[“f”]) and array elements (o[0], o[1], …) changes the butterfly.
- It also changes the structure ID.
- And possibly the indexing type.
- And possibly the m_offset field in the structure



- Adding or removing properties (o.f, o[“f”]) and array elements (o[0], o[1], …) changes the butterfly.
- It also changes the structure ID.
- And possibly the indexing type.
- And possibly the m_offset field in the structure
- *GC needs to read an atomic snapshot of these fields.*



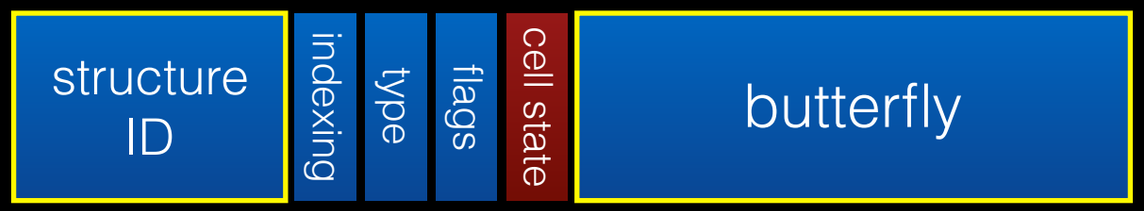
Simple Case



Goals:

- Allow mutator to set both fields without locks or atomic instructions.
- Allow collector to read both fields without locks or atomic instructions.

The Algorithm



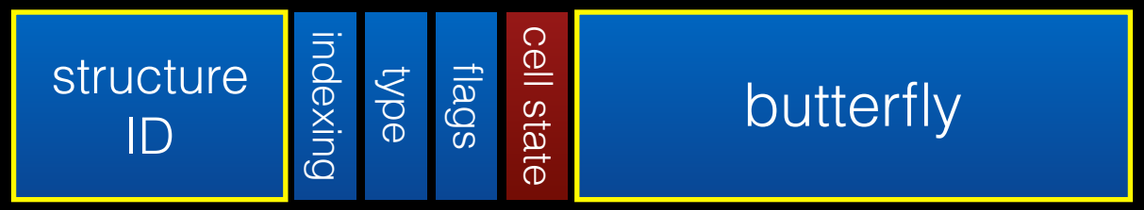
Mutator Algorithm

```
setStructureIDDirectly(
    nuke(structureID());
WTF::storeStoreFence();
m_butterfly.set(
    vm, this, newButterfly);
WTF::storeStoreFence();
setStructure(newStructure);
```

Collector Algorithm

```
StructureID early =
    structureID();
WTF::loadLoadFence();
Butterfly* butterfly =
    this->butterfly();
WTF::loadLoadFence();
StructureID late =
    structureID();
if (isNuked(early)
    || early != late)
    rescanLater();
```

The Algorithm



Mutator Algorithm

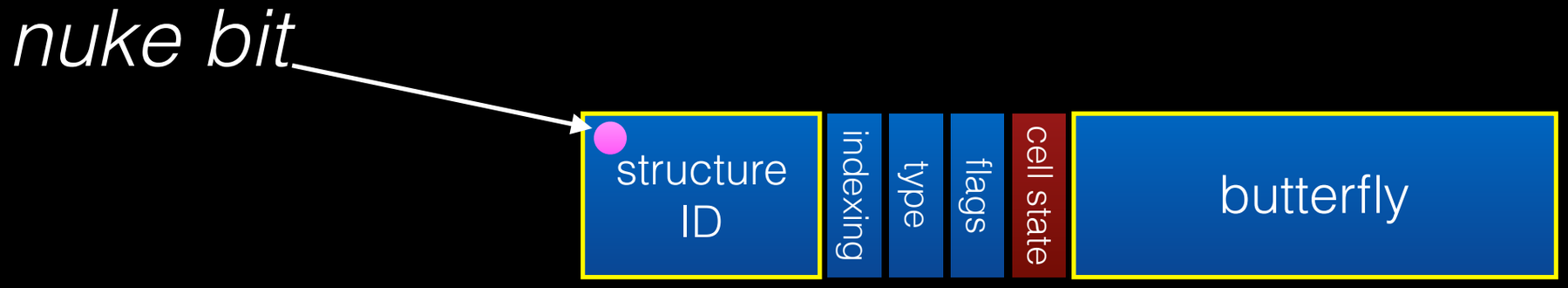
NukeStructure
StoreButterfly
StoreStructure

Collector Algorithm

LoadStructureEarly
LoadButterfly
LoadStructureLate

```
if (isNuked(early)
    || early != late)
    rescanLater();
```

The Algorithm



Mutator Algorithm

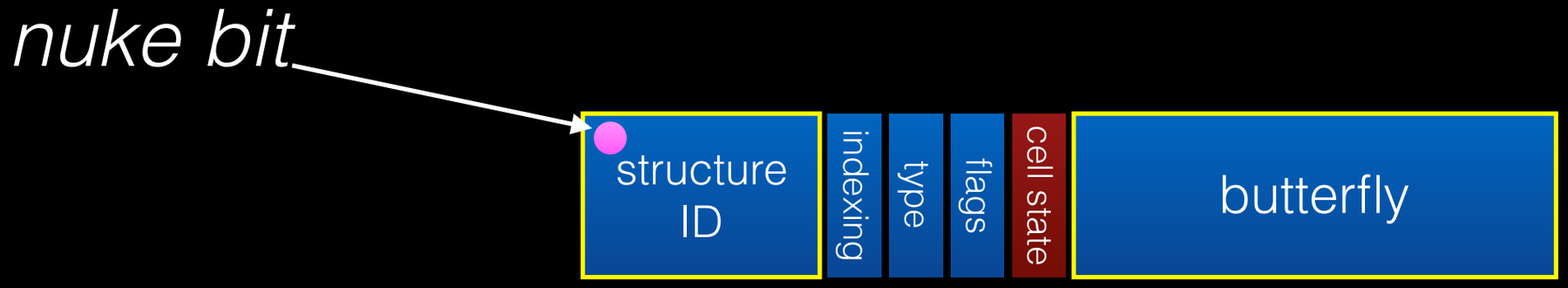
NukeStructure
StoreButterfly
StoreStructure

Collector Algorithm

LoadStructureEarly
LoadButterfly
LoadStructureLate

```
if (isNuked(early)
    || early != late)
    rescanLater();
```

The Algorithm



Mutator Algorithm

```
NukeStructure
StoreButterfly
StoreStructure
```

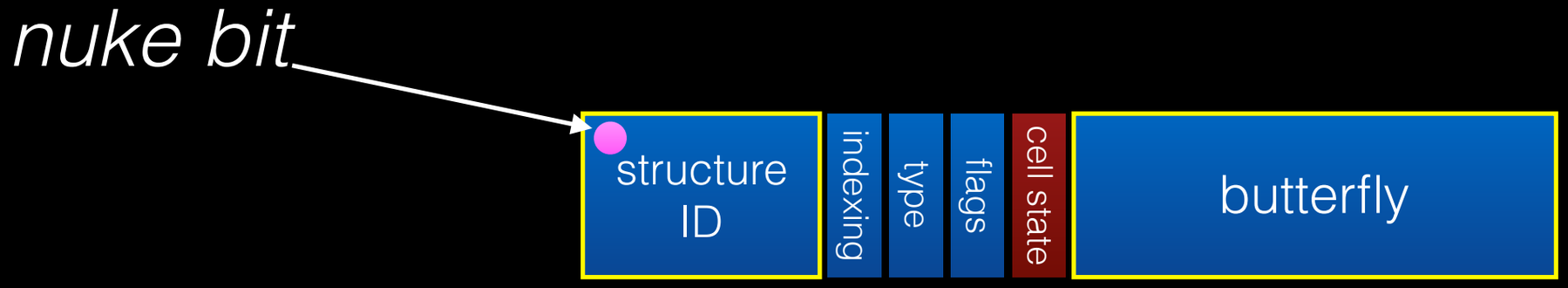
Collector Algorithm

```
LoadStructureEarly
LoadButterfly
LoadStructureLate

if (isNuked(early)
    || early != late)
    rescanLater();
```

Theorem: collector will rescan later if it sees a structure/butterfly mismatch.

The Algorithm



Mutator Algorithm

```
NukeStructure
StoreButterfly
StoreStructure
```

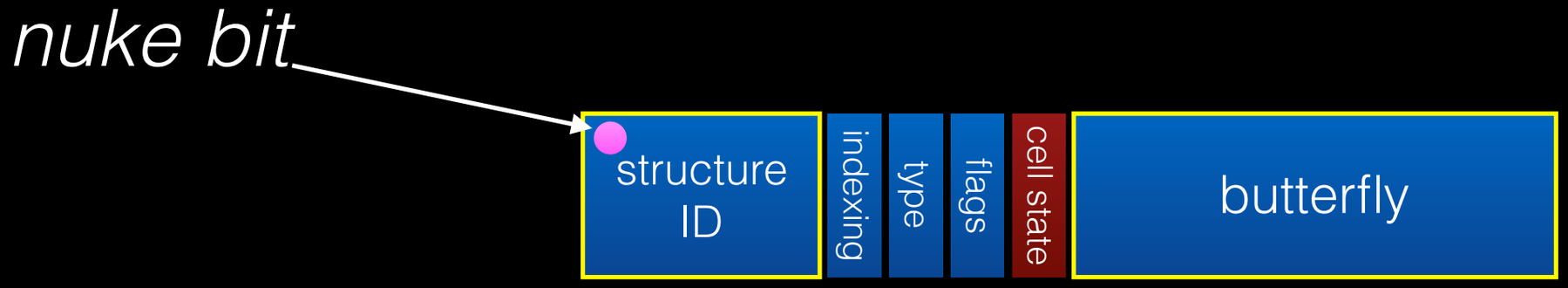
Collector Algorithm

```
LoadStructureEarly
LoadButterfly
LoadStructureLate

if (isNuked(early)
    || early != late)
    rescanLater();
```

Theorem: collector will rescan later if it sees a structure/butterfly mismatch.

The Algorithm



Mutator Algorithm

```
NukeStructure
StoreButterfly
StoreStructure
```

Collector Algorithm

```
LoadStructureEarly
LoadButterfly
LoadStructureLate

if (isNuked(early)
    || early != late)
    rescanLater();
```

Theorem: collector will rescan later if it sees a structure/butterfly mismatch.

Proof

NukeStructure StoreButterfly StoreStructure LoadStructureEarly LoadButterfly LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly StoreStructure LoadButterfly LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly LoadButterfly StoreStructure LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly LoadButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly StoreButterfly StoreStructure LoadButterfly LoadStructureLate
NukeStructure LoadStructureEarly StoreButterfly LoadButterfly StoreStructure LoadStructureLate
NukeStructure LoadStructureEarly StoreButterfly LoadButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly LoadButterfly StoreButterfly StoreStructure LoadStructureLate
NukeStructure LoadStructureEarly LoadButterfly StoreButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly LoadButterfly LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly NukeStructure StoreButterfly StoreStructure LoadButterfly LoadStructureLate
LoadStructureEarly NukeStructure StoreButterfly LoadButterfly StoreStructure LoadStructureLate
LoadStructureEarly NukeStructure StoreButterfly LoadButterfly LoadStructureLate StoreStructure
LoadStructureEarly NukeStructure LoadButterfly StoreButterfly StoreStructure LoadStructureLate
LoadStructureEarly NukeStructure LoadButterfly StoreButterfly LoadStructureLate StoreStructure
LoadStructureEarly NukeStructure LoadButterfly LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly LoadButterfly NukeStructure StoreButterfly StoreStructure LoadStructureLate
LoadStructureEarly LoadButterfly NukeStructure StoreButterfly LoadStructureLate StoreStructure
LoadStructureEarly LoadButterfly NukeStructure LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly LoadButterfly LoadStructureLate NukeStructure StoreButterfly StoreStructure

OK: collector sees new structure/butterfly

```
NukeStructure StoreButterfly StoreStructure LoadStructureEarly LoadButterfly LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly StoreStructure LoadButterfly LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly LoadButterfly StoreStructure LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly LoadButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly StoreButterfly StoreStructure LoadButterfly LoadStructureLate
NukeStructure LoadStructureEarly StoreButterfly LoadButterfly StoreStructure LoadStructureLate
NukeStructure LoadStructureEarly StoreButterfly LoadButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly LoadButterfly StoreButterfly StoreStructure LoadStructureLate
NukeStructure LoadStructureEarly LoadButterfly StoreButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly LoadButterfly LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly NukeStructure StoreButterfly StoreStructure LoadButterfly LoadStructureLate
LoadStructureEarly NukeStructure StoreButterfly LoadButterfly StoreStructure LoadStructureLate
LoadStructureEarly NukeStructure StoreButterfly LoadButterfly LoadStructureLate StoreStructure
LoadStructureEarly NukeStructure LoadButterfly StoreButterfly StoreStructure LoadStructureLate
LoadStructureEarly NukeStructure LoadButterfly StoreButterfly LoadStructureLate StoreStructure
LoadStructureEarly NukeStructure LoadButterfly LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly LoadButterfly NukeStructure StoreButterfly StoreStructure LoadStructureLate
LoadStructureEarly LoadButterfly NukeStructure StoreButterfly LoadStructureLate StoreStructure
LoadStructureEarly LoadButterfly NukeStructure LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly LoadButterfly LoadStructureLate NukeStructure StoreButterfly StoreStructure
```

OK: collector sees new structure/butterfly

```
NukeStructure StoreButterfly StoreStructure LoadStructureEarly LoadButterfly LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly StoreStructure LoadButterfly LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly LoadButterfly StoreStructure LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly LoadButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly StoreButterfly StoreStructure LoadButterfly LoadStructureLate
NukeStructure LoadStructureEarly StoreButterfly LoadButterfly StoreStructure LoadStructureLate
NukeStructure LoadStructureEarly StoreButterfly LoadButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly LoadButterfly StoreButterfly StoreStructure LoadStructureLate
NukeStructure LoadStructureEarly LoadButterfly StoreButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly LoadButterfly LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly NukeStructure StoreButterfly StoreStructure LoadButterfly LoadStructureLate
LoadStructureEarly NukeStructure StoreButterfly LoadButterfly StoreStructure LoadStructureLate
LoadStructureEarly NukeStructure StoreButterfly LoadButterfly LoadStructureLate StoreStructure
LoadStructureEarly NukeStructure LoadButterfly StoreButterfly StoreStructure LoadStructureLate
LoadStructureEarly NukeStructure LoadButterfly StoreButterfly LoadStructureLate StoreStructure
LoadStructureEarly NukeStructure LoadButterfly LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly LoadButterfly NukeStructure StoreButterfly StoreStructure LoadStructureLate
LoadStructureEarly LoadButterfly NukeStructure StoreButterfly LoadStructureLate StoreStructure
LoadStructureEarly LoadButterfly NukeStructure LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly LoadButterfly LoadStructureLate NukeStructure StoreButterfly StoreStructure
```

OK: collector sees old structure/butterfly

OK: collector sees new structure/butterfly

NukeStructure StoreButterfly StoreStructure LoadStructureEarly LoadButterfly LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly StoreStructure LoadButterfly LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly LoadButterfly StoreStructure LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly LoadButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly StoreButterfly StoreStructure LoadButterfly LoadStructureLate
NukeStructure LoadStructureEarly StoreButterfly LoadButterfly StoreStructure LoadStructureLate
NukeStructure LoadStructureEarly StoreButterfly LoadButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly LoadButterfly StoreButterfly StoreStructure LoadStructureLate
NukeStructure LoadStructureEarly LoadButterfly StoreButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly LoadButterfly LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly NukeStructure StoreButterfly StoreStructure LoadButterfly LoadStructureLate
LoadStructureEarly NukeStructure StoreButterfly LoadButterfly StoreStructure LoadStructureLate
LoadStructureEarly NukeStructure StoreButterfly LoadButterfly LoadStructureLate StoreStructure
LoadStructureEarly NukeStructure LoadButterfly StoreButterfly StoreStructure LoadStructureLate
LoadStructureEarly NukeStructure LoadButterfly StoreButterfly LoadStructureLate StoreStructure
LoadStructureEarly NukeStructure LoadButterfly LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly LoadButterfly NukeStructure StoreButterfly StoreStructure LoadStructureLate
LoadStructureEarly LoadButterfly NukeStructure StoreButterfly LoadStructureLate StoreStructure
LoadStructureEarly LoadButterfly NukeStructure LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly LoadButterfly LoadStructureLate NukeStructure StoreButterfly StoreStructure

Rescan!

OK: collector sees old structure/butterfly

OK: collector sees new structure/butterfly

NukeStructure StoreButterfly StoreStructure LoadStructureEarly LoadButterfly LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly StoreStructure LoadButterfly LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly LoadButterfly StoreStructure LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly LoadButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly StoreButterfly StoreStructure LoadButterfly LoadStructureLate
NukeStructure LoadStructureEarly StoreButterfly LoadButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly LoadButterfly StoreButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly LoadButterfly StoreButterfly StoreStructure LoadStructureLate
LoadStructureEarly NukeStructure StoreButterfly StoreStructure LoadButterfly LoadStructureLate
LoadStructureEarly NukeStructure StoreButterfly LoadButterfly StoreStructure LoadStructureLate
LoadStructureEarly NukeStructure StoreButterfly LoadButterfly LoadStructureLate StoreStructure
LoadStructureEarly NukeStructure LoadButterfly StoreButterfly StoreStructure LoadStructureLate
LoadStructureEarly NukeStructure LoadButterfly StoreButterfly LoadStructureLate StoreStructure
LoadStructureEarly NukeStructure LoadButterfly LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly LoadButterfly NukeStructure StoreButterfly StoreStructure LoadStructureLate
LoadStructureEarly LoadButterfly NukeStructure StoreButterfly LoadStructureLate StoreStructure
LoadStructureEarly LoadButterfly NukeStructure LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly LoadButterfly LoadStructureLate NukeStructure StoreButterfly StoreStructure

Rescan!

OK: collector sees old structure/butterfly

OK: collector sees new structure/butterfly

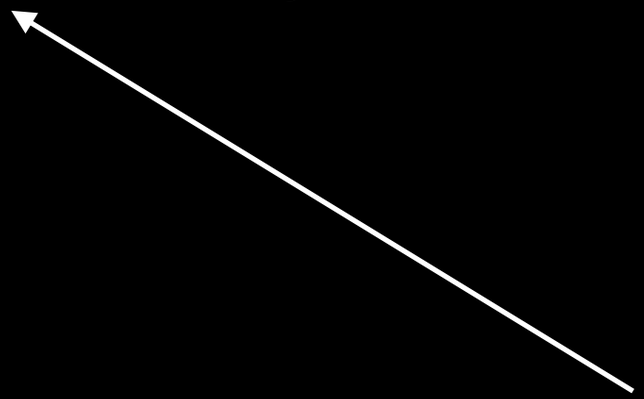
NukeStructure StoreButterfly StoreStructure LoadStructureEarly LoadButterfly LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly StoreStructure LoadButterfly LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly LoadButterfly StoreStructure LoadStructureLate
NukeStructure StoreButterfly LoadStructureEarly LoadButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly StoreButterfly StoreStructure LoadButterfly LoadStructureLate
NukeStructure LoadStructureEarly StoreButterfly LoadButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly LoadButterfly StoreButterfly LoadStructureLate StoreStructure
NukeStructure LoadStructureEarly LoadButterfly StoreButterfly StoreStructure LoadStructureLate
NukeStructure LoadStructureEarly LoadButterfly LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly NukeStructure StoreButterfly StoreStructure LoadButterfly LoadStructureLate
LoadStructureEarly NukeStructure StoreButterfly LoadButterfly StoreStructure LoadStructureLate
LoadStructureEarly NukeStructure LoadButterfly StoreButterfly StoreStructure LoadStructureLate
LoadStructureEarly NukeStructure LoadButterfly LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly LoadButterfly NukeStructure StoreButterfly StoreStructure LoadStructureLate
LoadStructureEarly LoadButterfly NukeStructure StoreButterfly LoadStructureLate StoreStructure
LoadStructureEarly LoadButterfly NukeStructure LoadStructureLate StoreButterfly StoreStructure
LoadStructureEarly LoadButterfly LoadStructureLate NukeStructure StoreButterfly StoreStructure

Rescan!

OK: collector sees old structure/butterfly

NukeStructure LoadStructureEarly StoreButterfly LoadButterfly LoadStructureLate StoreStructure

```
NukeStructure LoadStructureEarly StoreButterfly LoadButterfly LoadStructureLate StoreStructure
```



LoadStructureEarly sees a nuked structure

LoadStructureEarly NukeStructure LoadButterfly StoreButterfly StoreStructure LoadStructureLate

```
LoadStructureEarly NukeStructure LoadButterfly StoreButterfly StoreStructure LoadStructureLate
```

LoadStructureEarly and LoadStructureLate see different structures

Obstruction-free

“Double Collect Snapshot”

“Atomic Snapshots of Shared Memory”
Afek, Attiya, Dolev, Gafni, Merritt, and Shavit

Obstruction-free Double Collect Snapshot

- Snapshotting the *indexing type* and *m_offset* is harder but follows the same principle.
- We avoid fences in the mutator when the GC is not running.
- We completely avoid fences in the collector.
- See JSObject::visitButterflyImpl in JSObject.cpp.
- Too hard to use for all objects…

Riptide

- Introduction
- Retreating Wavefront
- Space-Time Scheduler
- Safepoints
- Obstruction-Free Double Collect Snapshot
- Embedded WTF Locks
- Conclusion

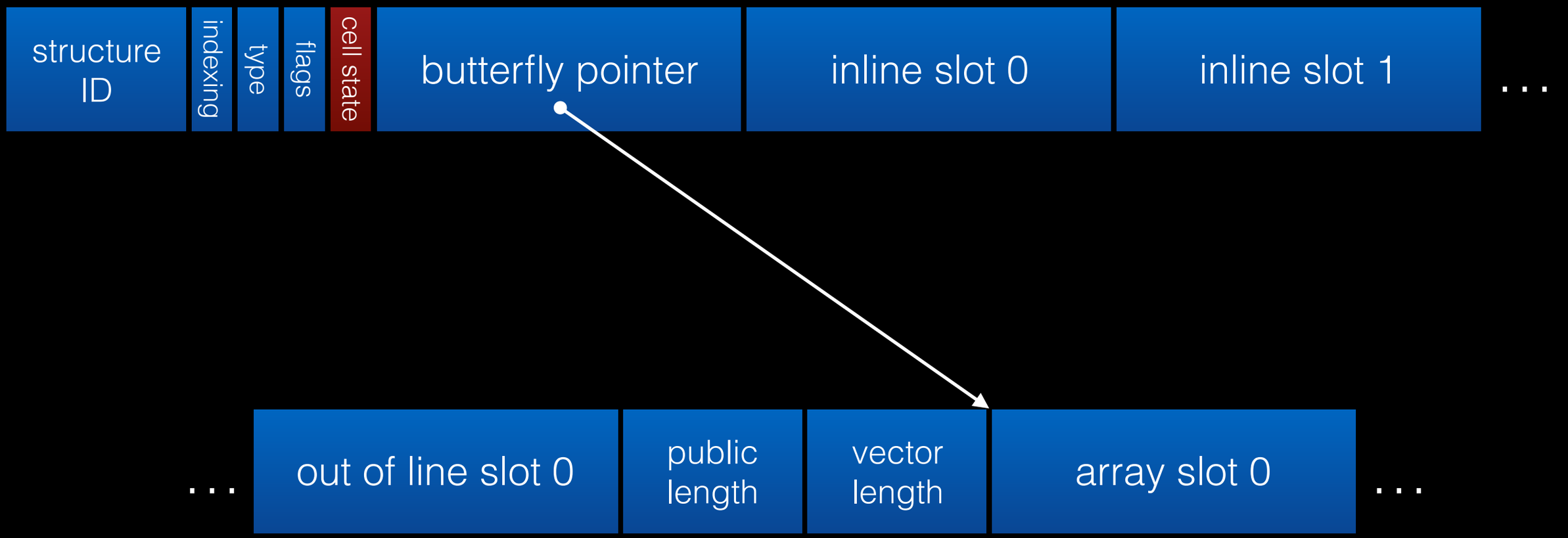
- We support many kinds of exotic objects.
- They are implemented in C++ with some JIT inlining.
- They may reshape themselves in ways that race with the GC.

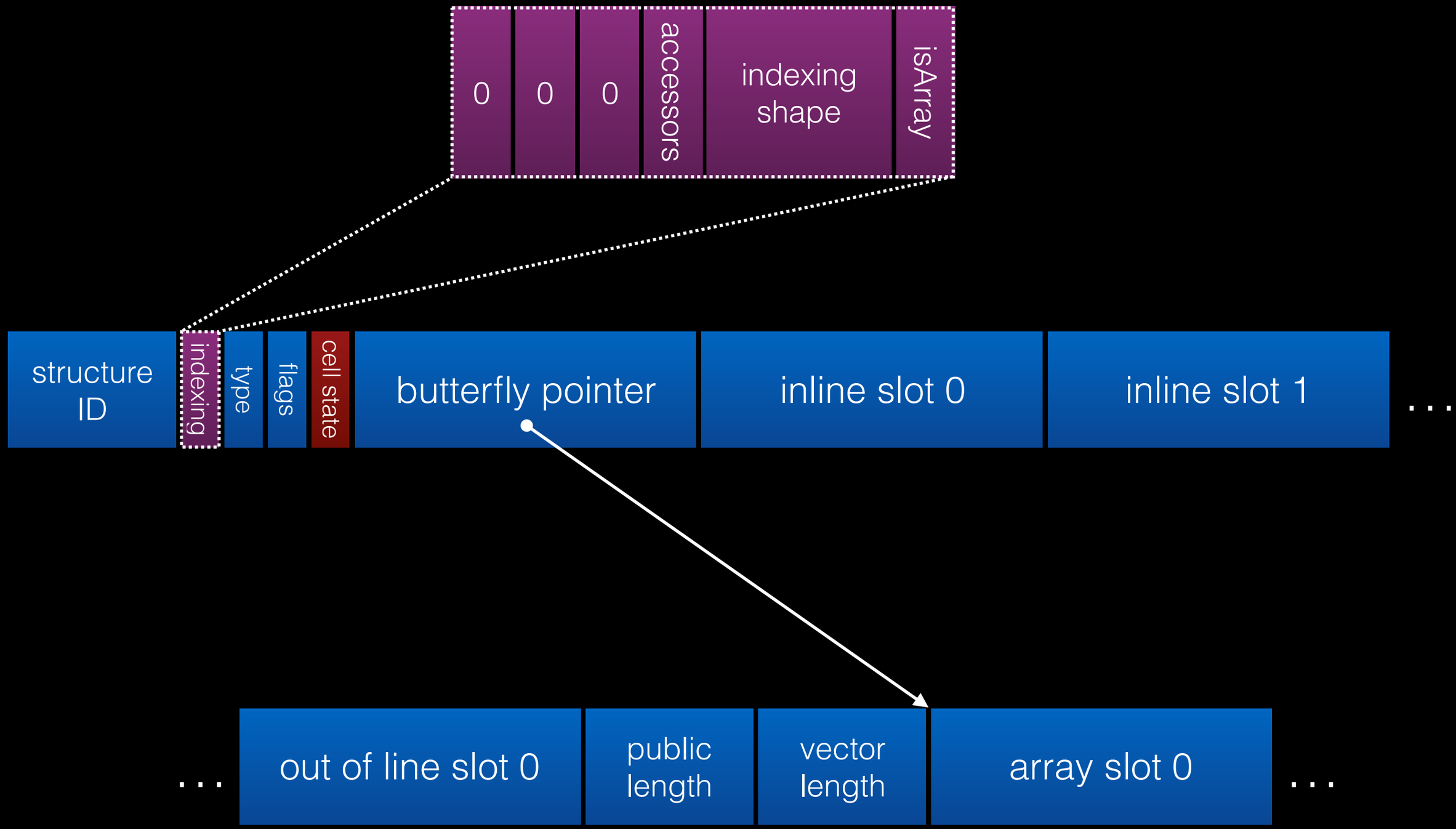
Locks!

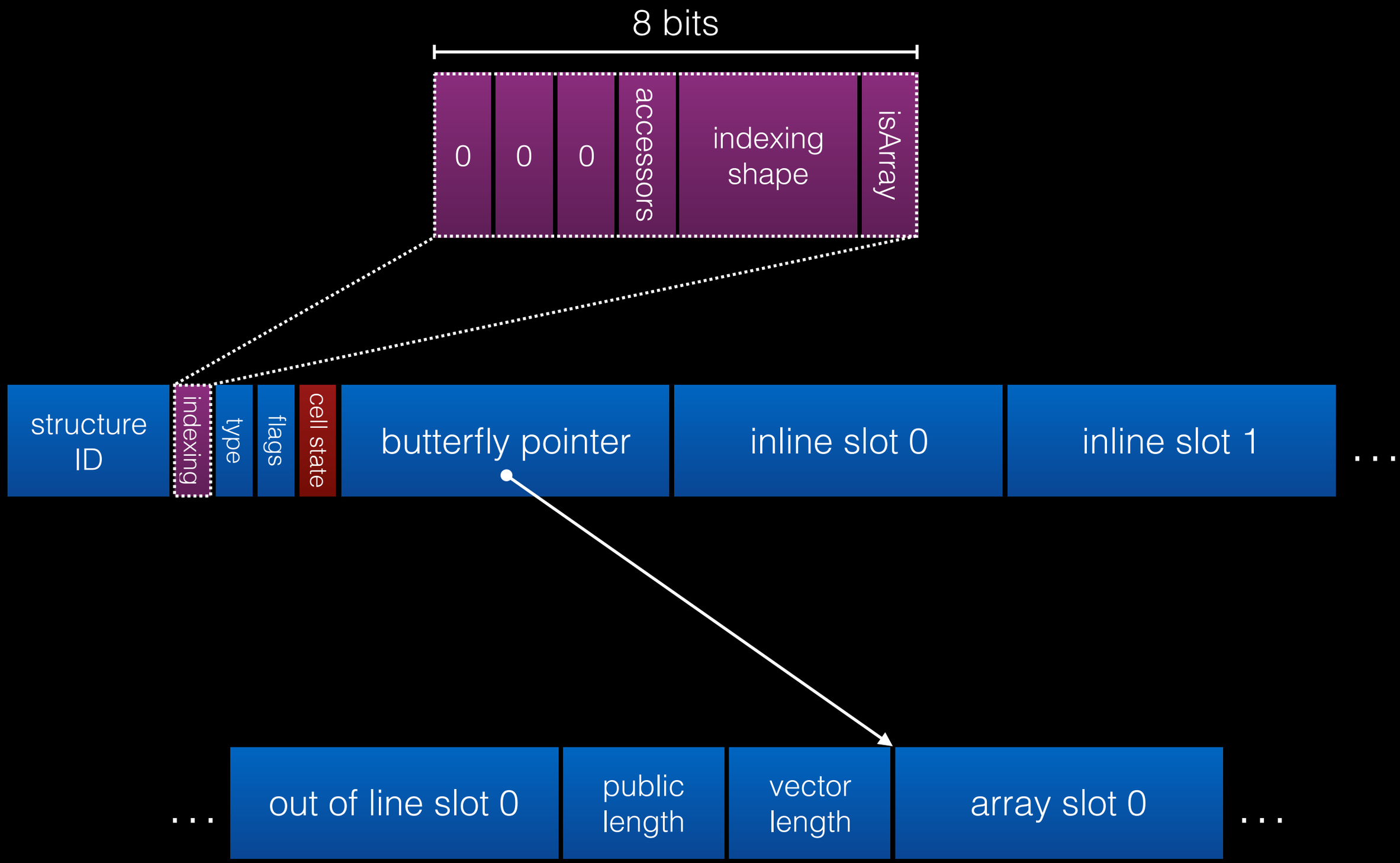
- Every object has a fully adaptive internal lock hidden in its header.

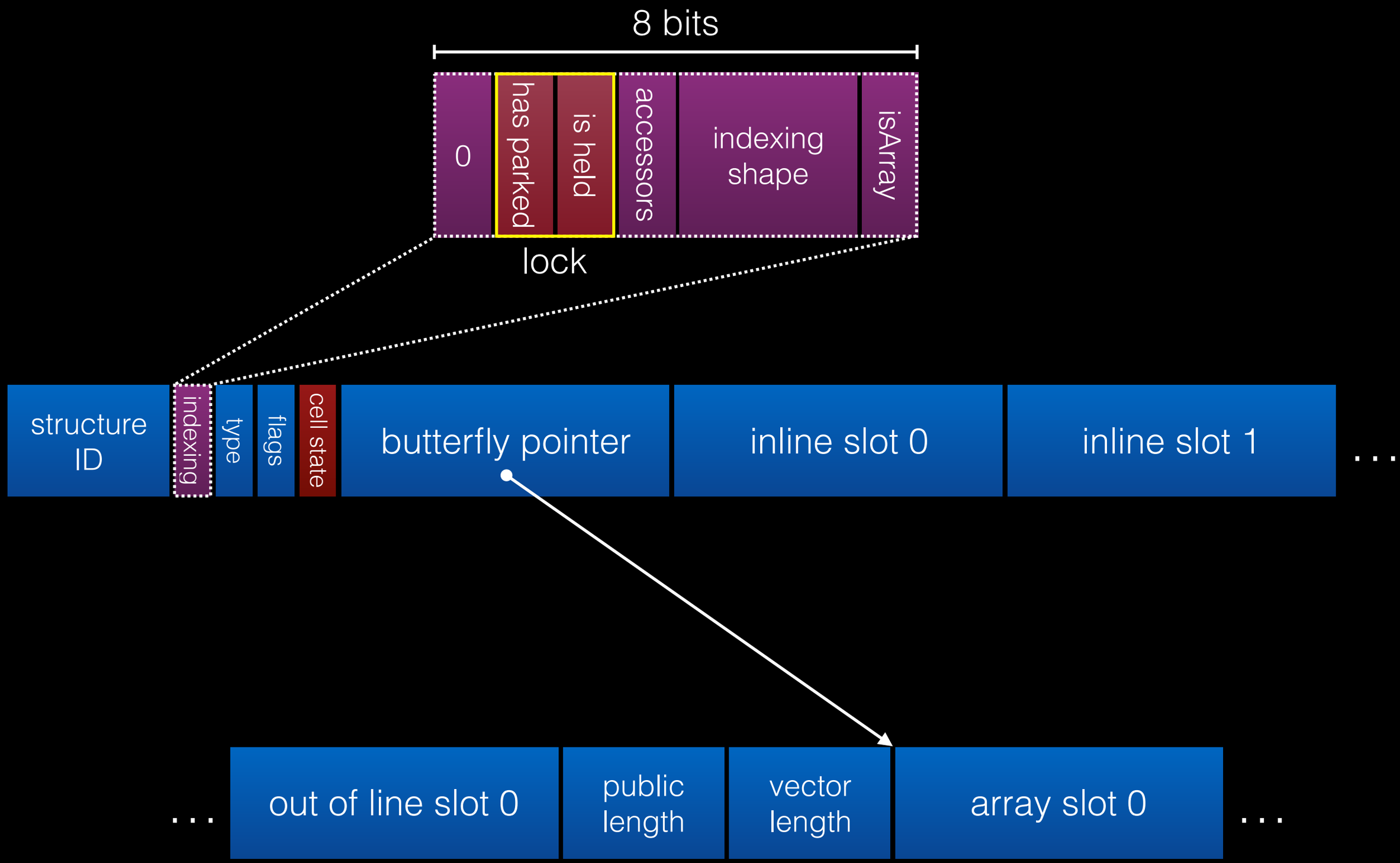
Internal Object Lock

- Inline single-CAS fast path for locking.
- Inline single-CAS fast path for unlocking.
- No unbounded spinning.
- Stochastically fair
  - *converges to FIFO as critical section time approaches 1ms*


structure ID
indexing
type
flags
cell state
butterfly pointer
inline slot 0
inline slot 1
...
...
out of line slot 0
public length
vector length
array slot 0
...


0
0
0
accessors
indexing shape
isArray
structure ID
indexing
type
flags
cell state
butterfly pointer
inline slot 0
inline slot 1
...
...
out of line slot 0
public length
vector length
array slot 0
...


8 bits
0
0
0
accessors
indexing shape
isArray
structure ID
indexing
type
flags
cell state
butterfly pointer
inline slot 0
inline slot 1
...
...
out of line slot 0
public length
vector length
array slot 0
...


8 bits
0
has parked
is held
accessors
indexing shape
isArray
lock
structure ID
indexing
type
flags
cell state
butterfly pointer
inline slot 0
inline slot 1
...
...
out of line slot 0
public length
vector length
array slot 0
...

Internal Object Lock

- Used for:
  - Array resizing
  - Sparse maps
  - Code blocks
  - Property name enumerator

Riptide

- Introduction
- Retreating Wavefront
- Space-Time Scheduler
- Safepoints
- Obstruction-Free Double Collect Snapshot
- Embedded WTF Locks
- Conclusion

What Riptide Does For WebKit

- Same throughput as our old collector.
  - Page load times, wide variety of benchmark suites.
- Same memory usage as our old collector.
  - Membuster, PLUM (internal tests)
- Much lower latency than our previous collector.
  - Much higher score on Octane/SplayLatency

Agenda

- ~~Introduction~~
- ~~JavaScriptCore~~
- ~~Efficient Mark-Sweep~~
  - *30 minute break*
- ~~Concurrent GC~~
- bmalloc
- WTF::Lock