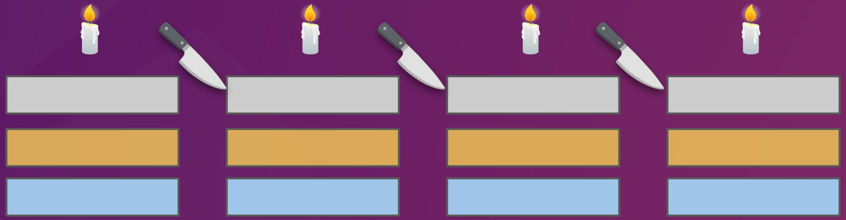
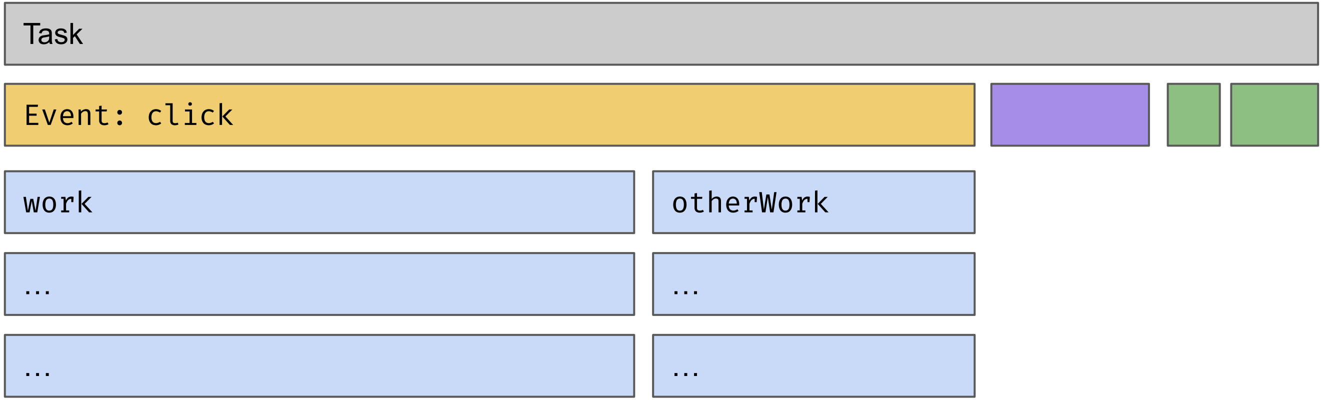
Cut My Task Into Pieces

This is Concurrent Mode



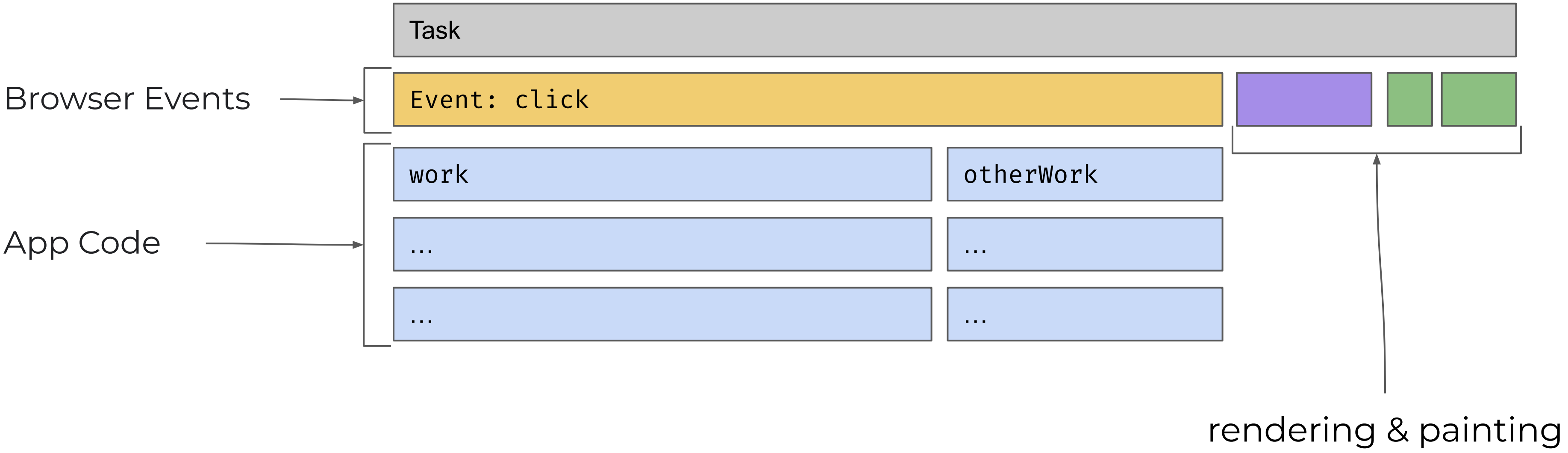
What happens when users interact with your app?

A task



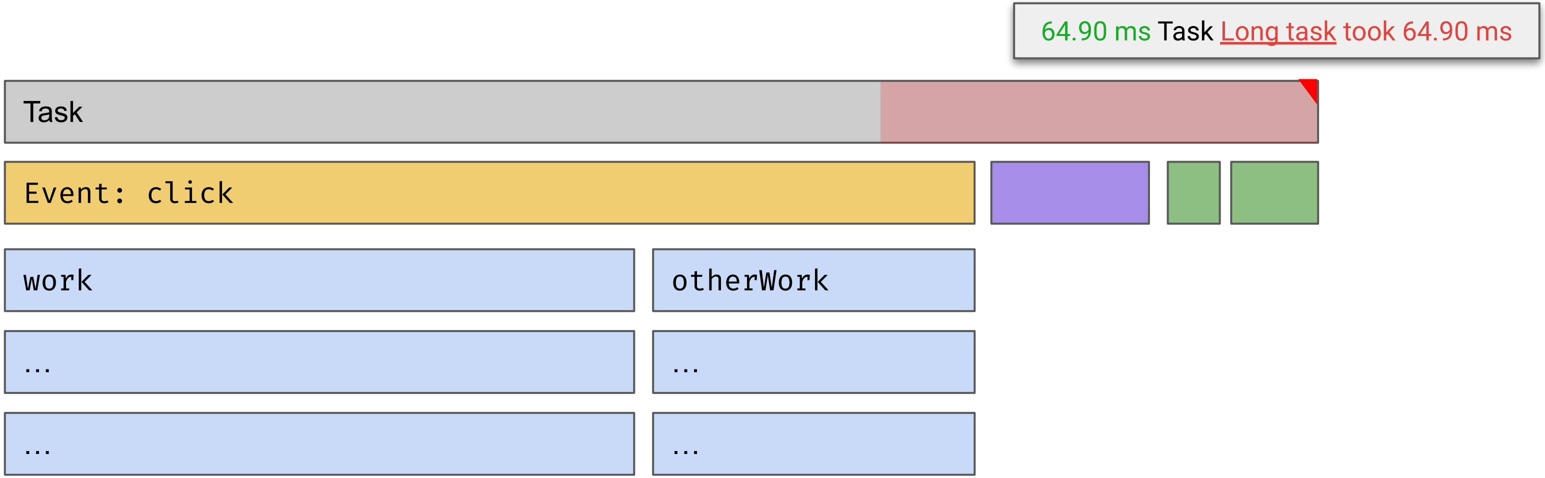
A task

any discrete piece of work that the browser does



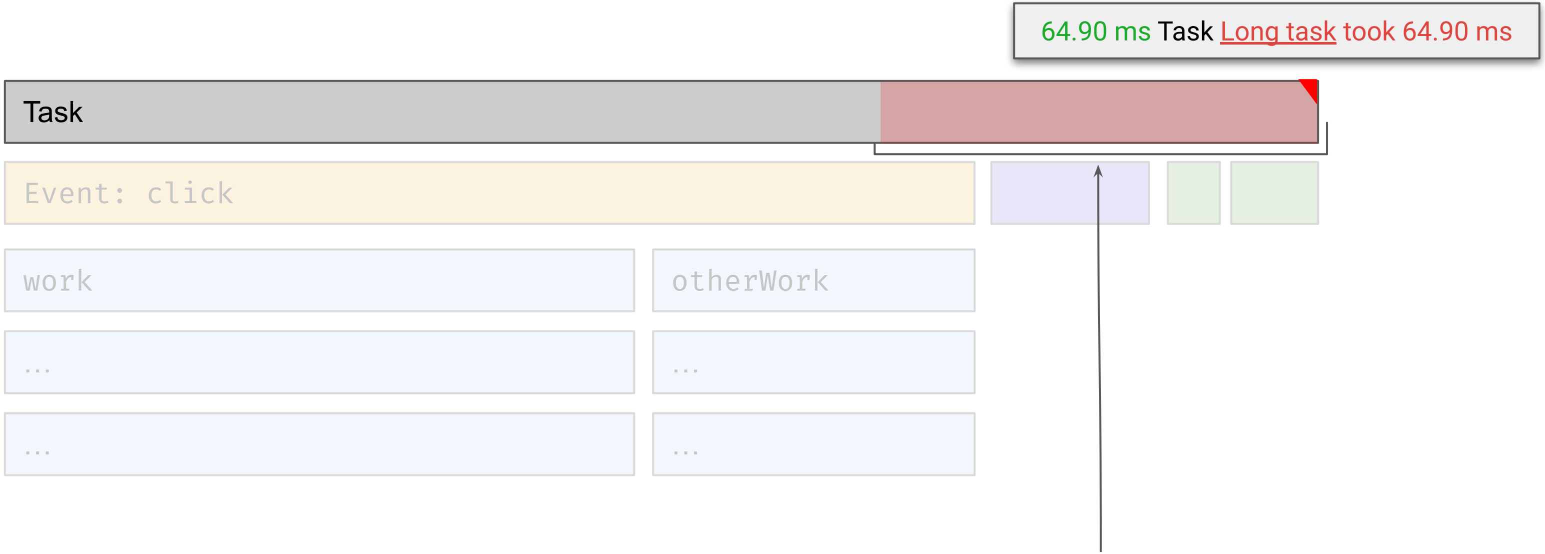
... and more

A long task



a task that exceeds 50ms processing time

A long task



64.90 ms Task Long task took 64.90 ms

Task

Event: click

work

otherWork

...

...

...

...

blocking time: 14.9ms

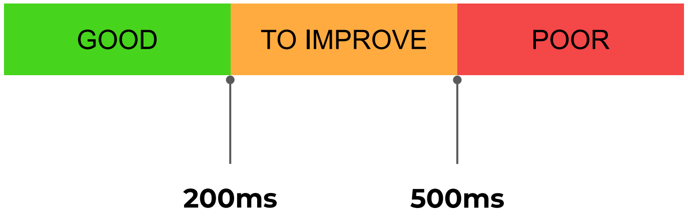
LOOKS SCARY

😱

Why should you care?

INP

Interaction to Next Paint



GOOD

TO IMPROVE

POOR

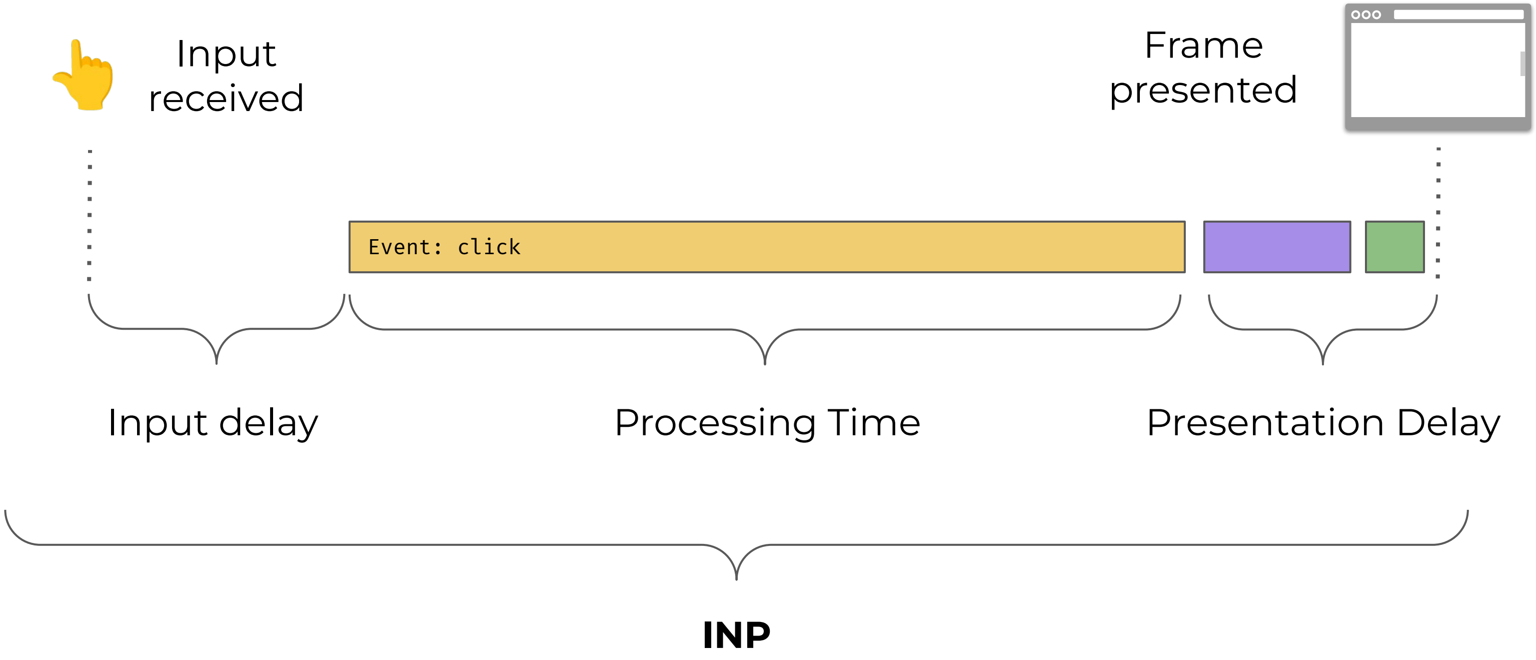
200ms

500ms

https://web.dev/inp/

Core Web Vital: INP (replaced FID at 12.03.24)

INP


Input received
Frame presented
Event: click
Input delay
Processing Time
Presentation Delay
INP

Julian Jandl

Performance Engineer

Trainer & Consultant

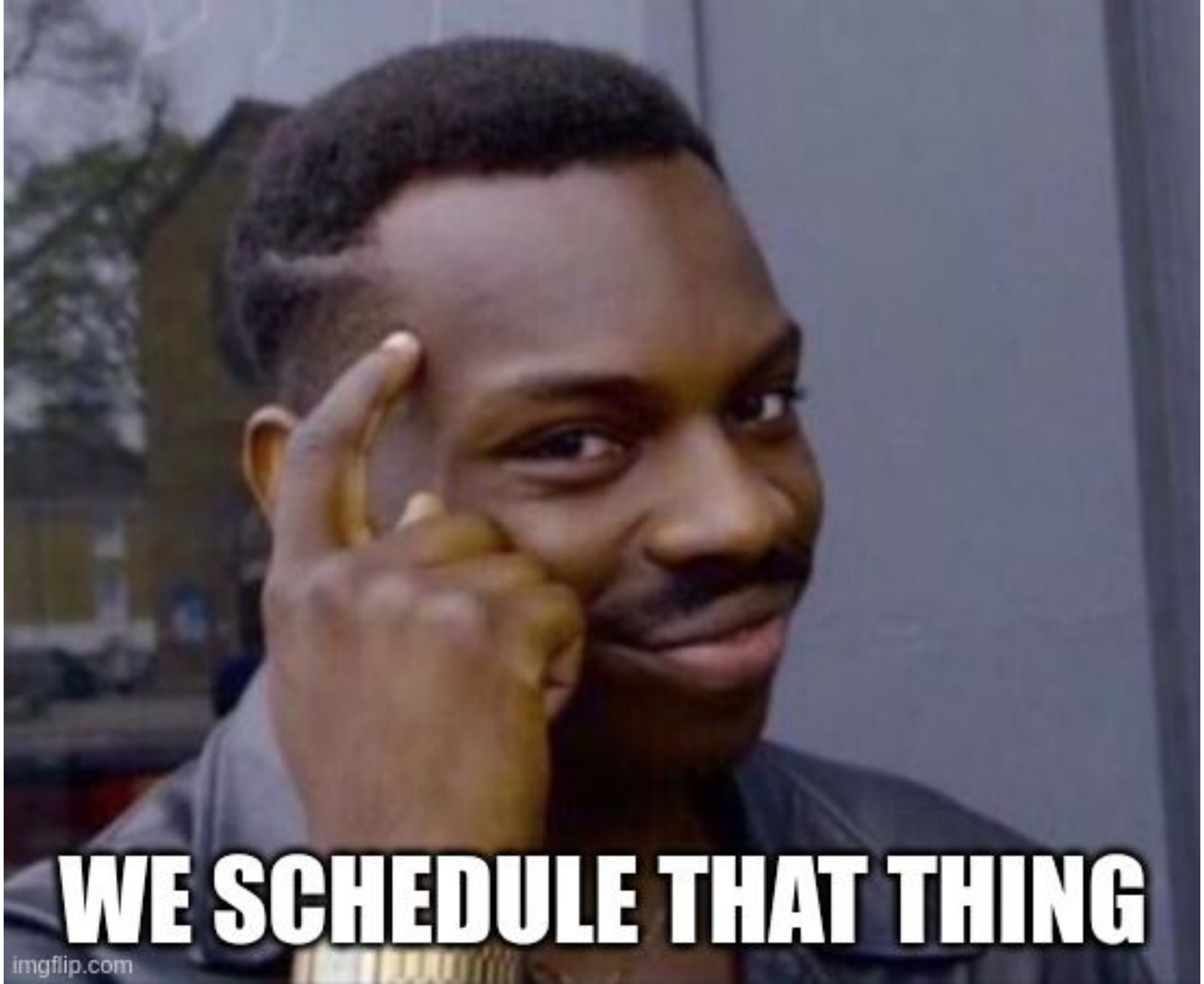
@rx-angular core maintainer

@hoebbelsB

Live Demo

Long Task & INP

What options do we have?


WE SCHEDULE THAT THING
imgflip.com

Scheduling Types: Overview

Timer

setTimeout, setInterval

IdleCallback

requestIdleCallback

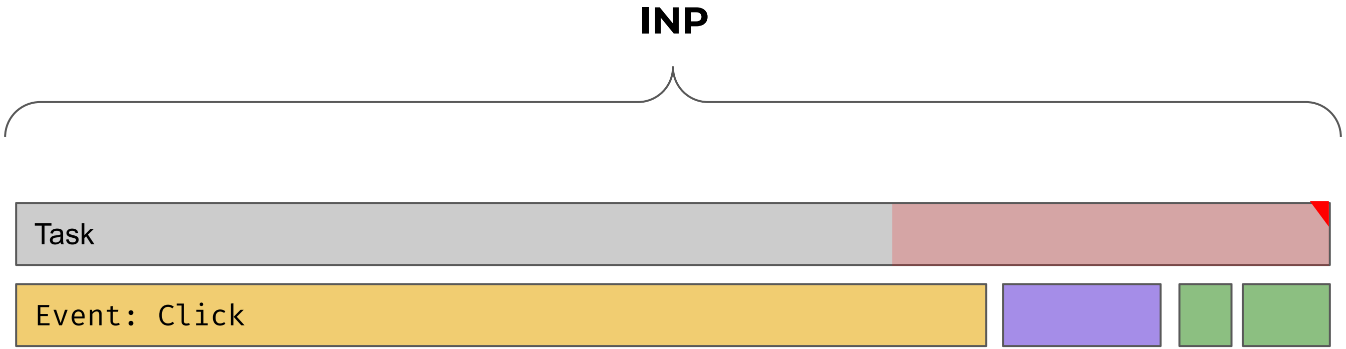
postTask

Scheduler.postTask

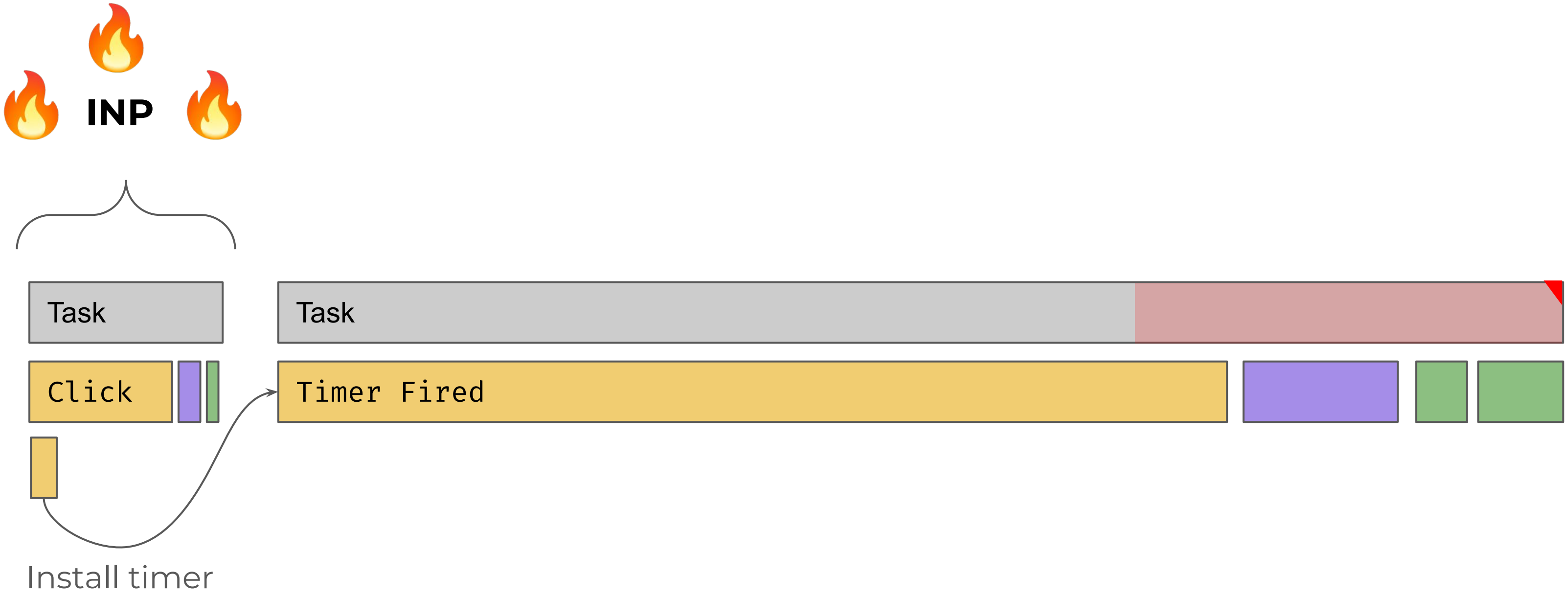
MessageChannel

x-tab comm.

Long task affecting INP



Schedule a long task for better INP



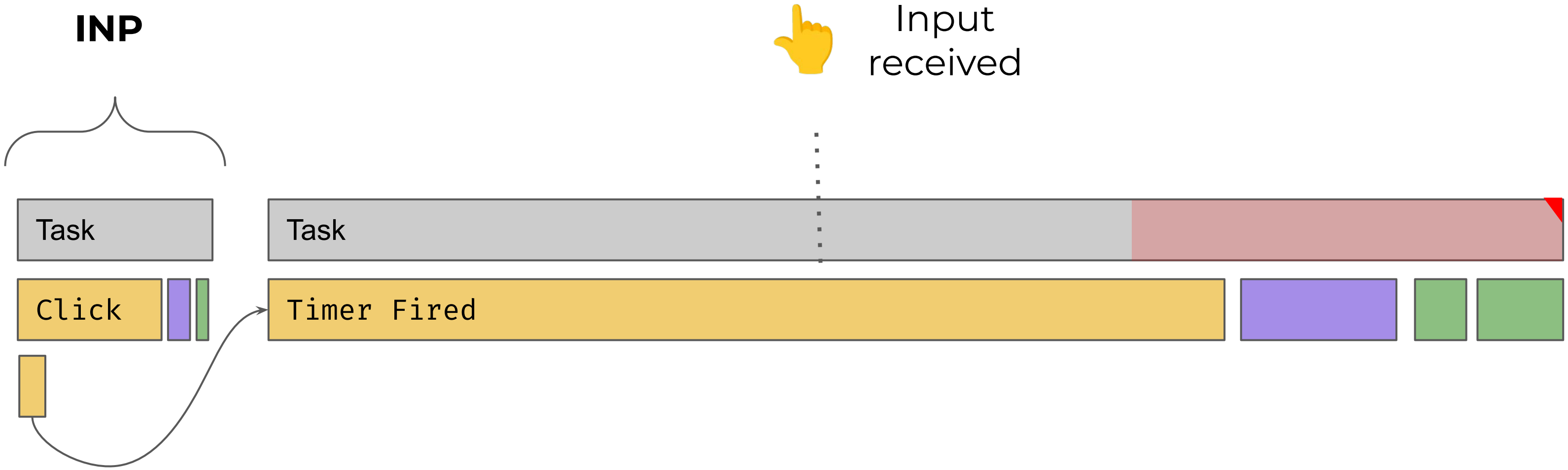
Live Demo

Scheduling to fix INP

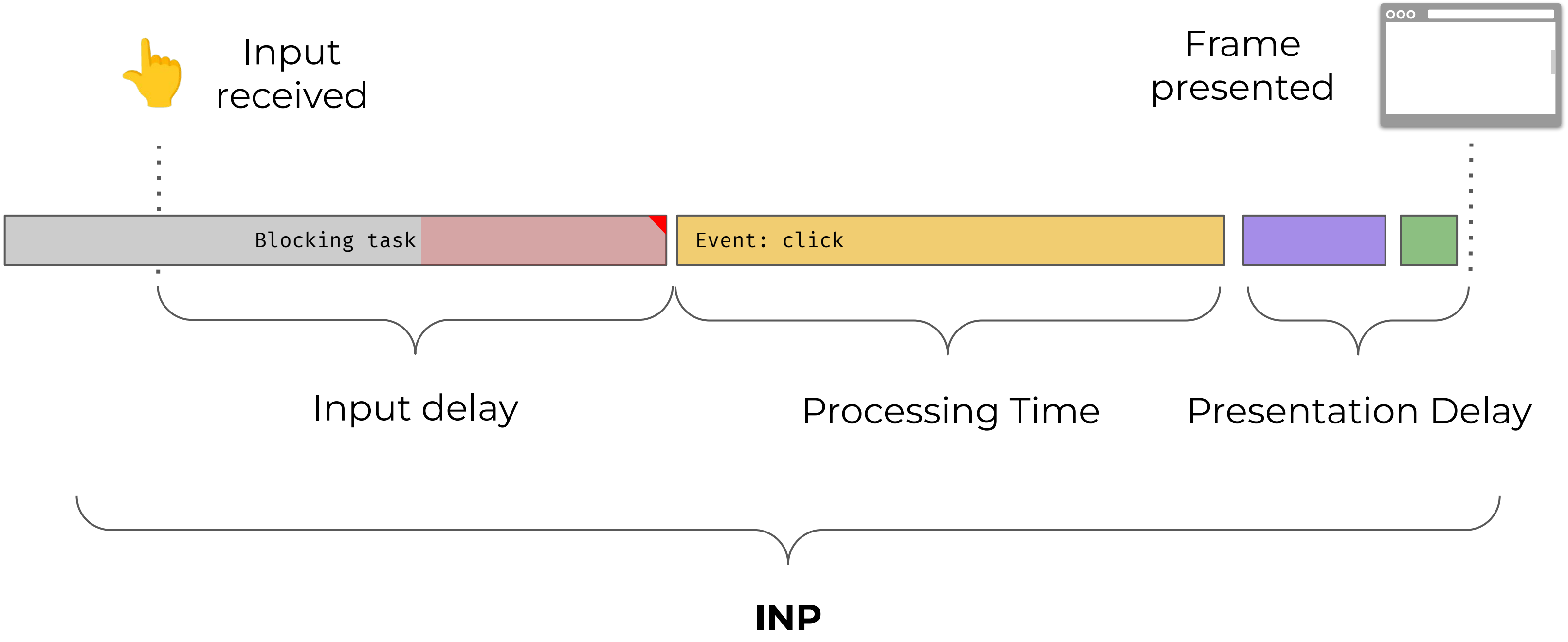
🥳 Yes this works! 🔥

🚧 BUT 😱

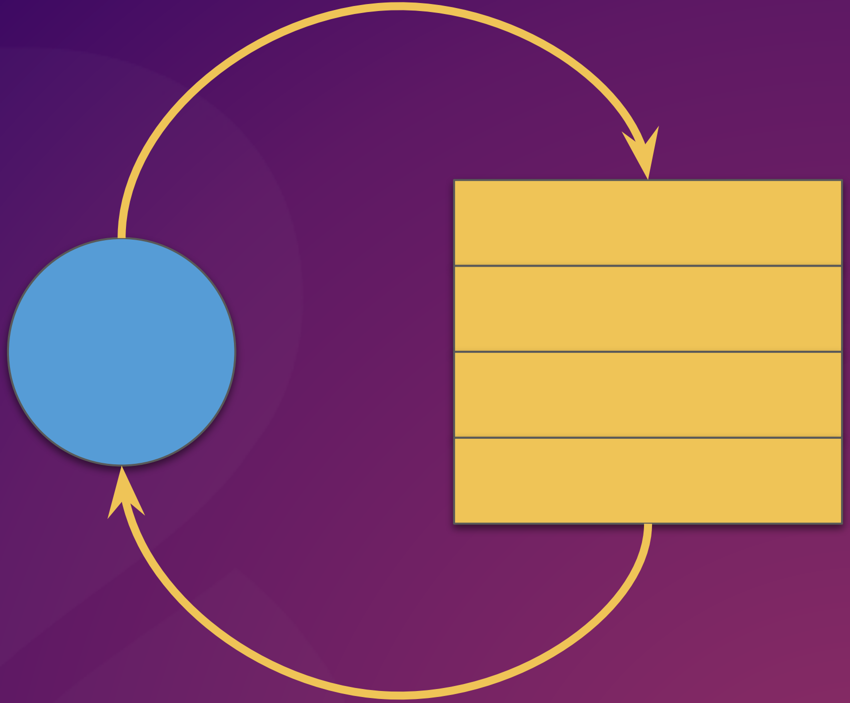
Schedule a long task for “better” INP



INP

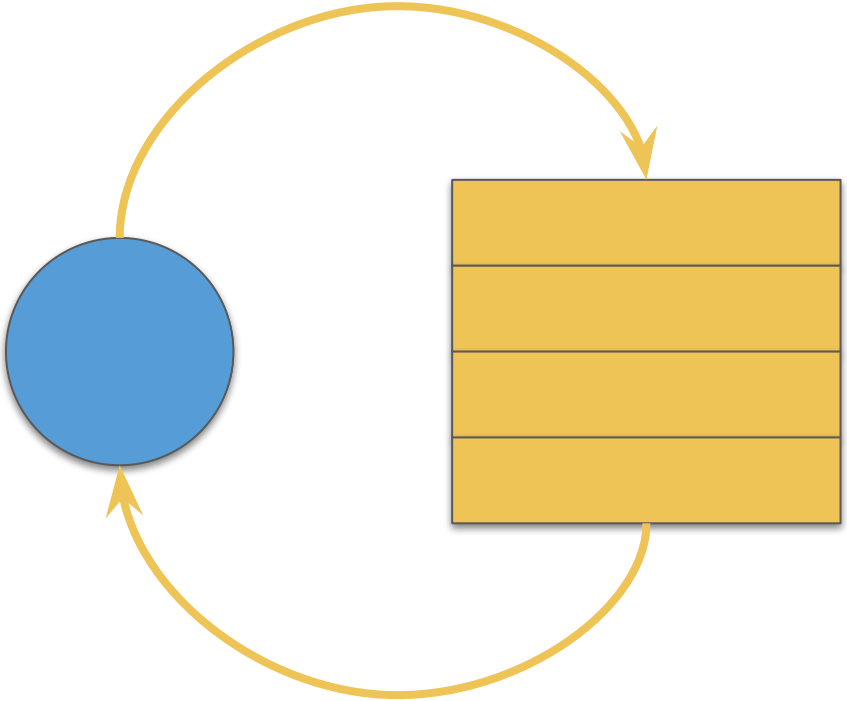

Input received
Frame presented
Blocking task
Event: click
Input delay
Processing Time
Presentation Delay
INP

The Event Loop



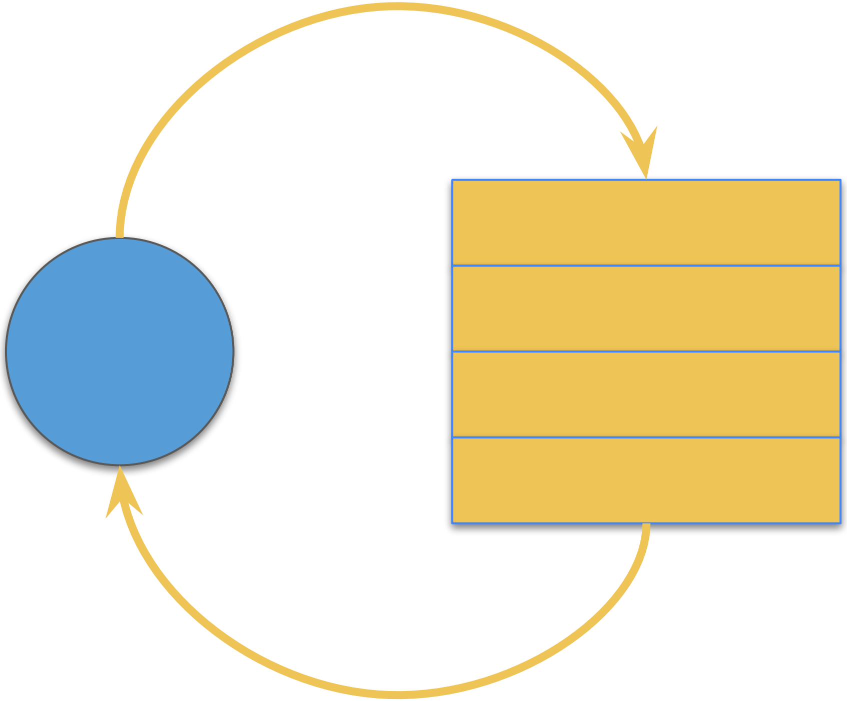
https://javascript.info/event-loop

The Event Loop



```
while (queue.waitForMessage()){
  queue.processNextMessage();
}
```

The Event Loop



💡 The macrotask queue

```
while (queue.waitForMessage()){
  queue.processNextMessage();
}
```

Macrotask - overview of macrotasks

Timer

`setTimeout, setInterval`

IdleCallback

`requestIdleCallback`

MessageChannel

`postTask, x-tab comm.`

rAF

`requestAnimationFrame`

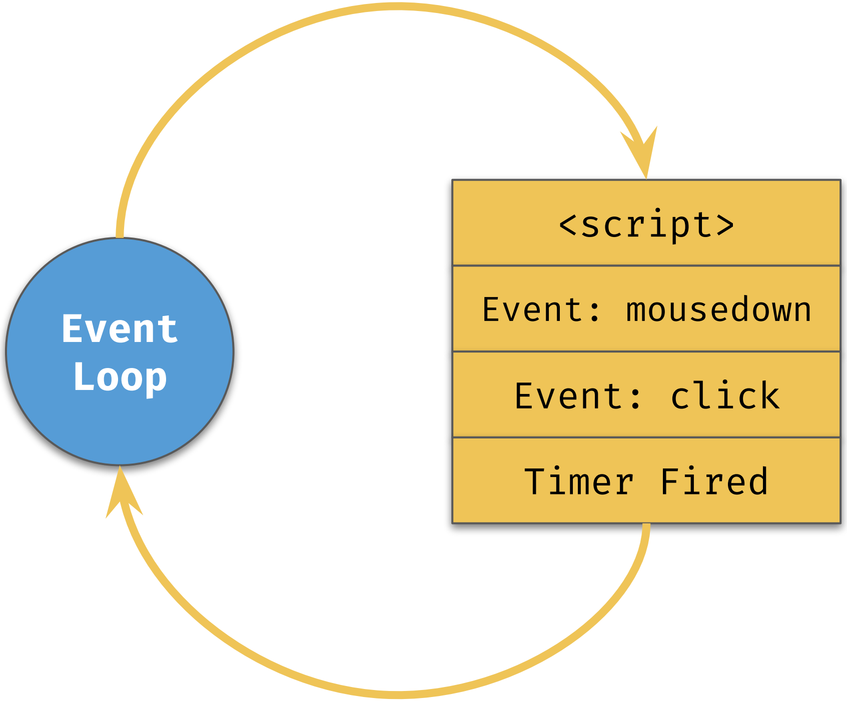
Browser / DOM Events

```
click, drag, resize,
http, script eval, ...
```

3rd parties & more

`Browser extensions, ...`

The Event Loop

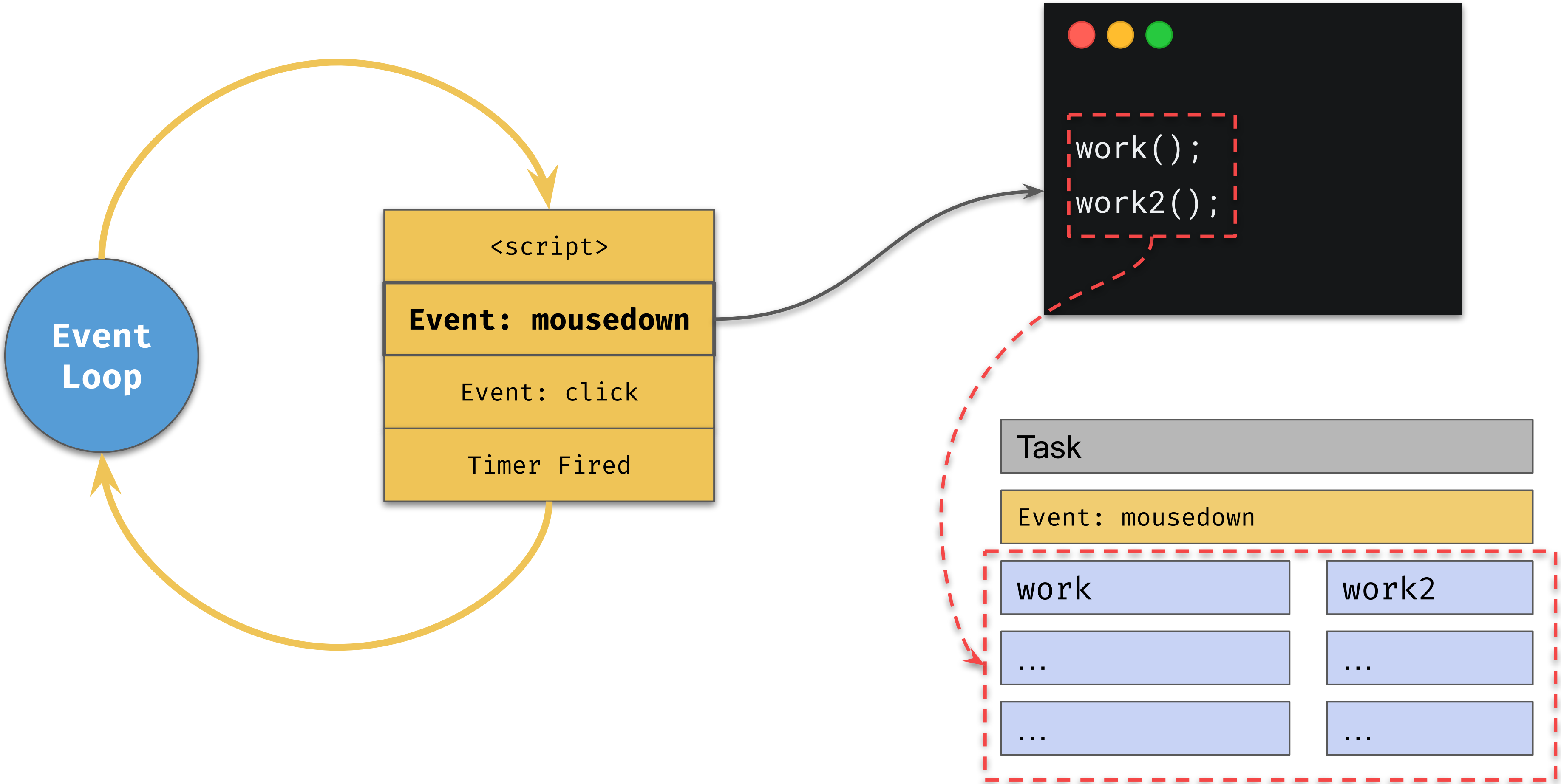


Macrotask
Queue

"Run-to-completion"

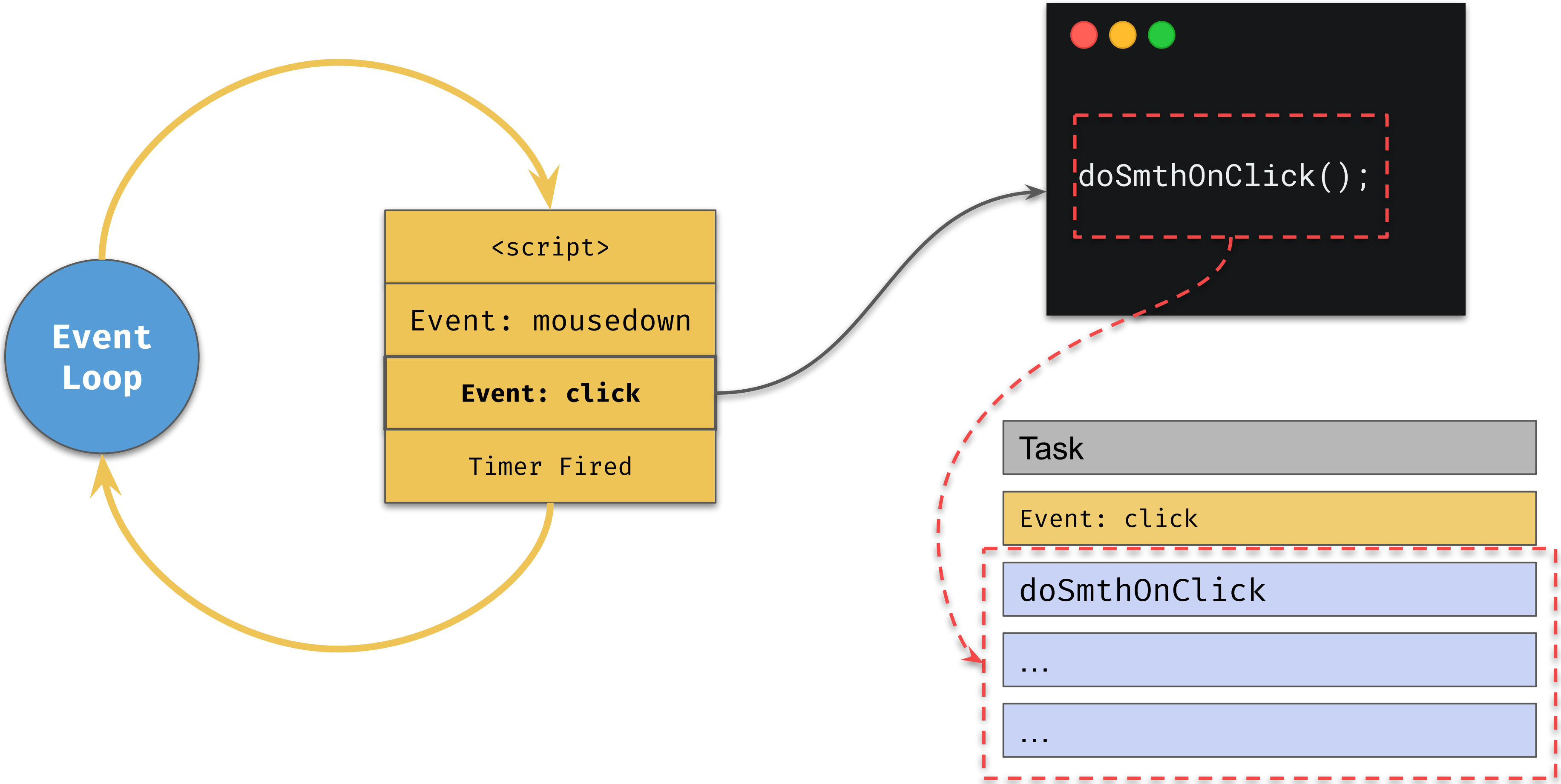
https://javascript.info/event-loop

Macrotask execution



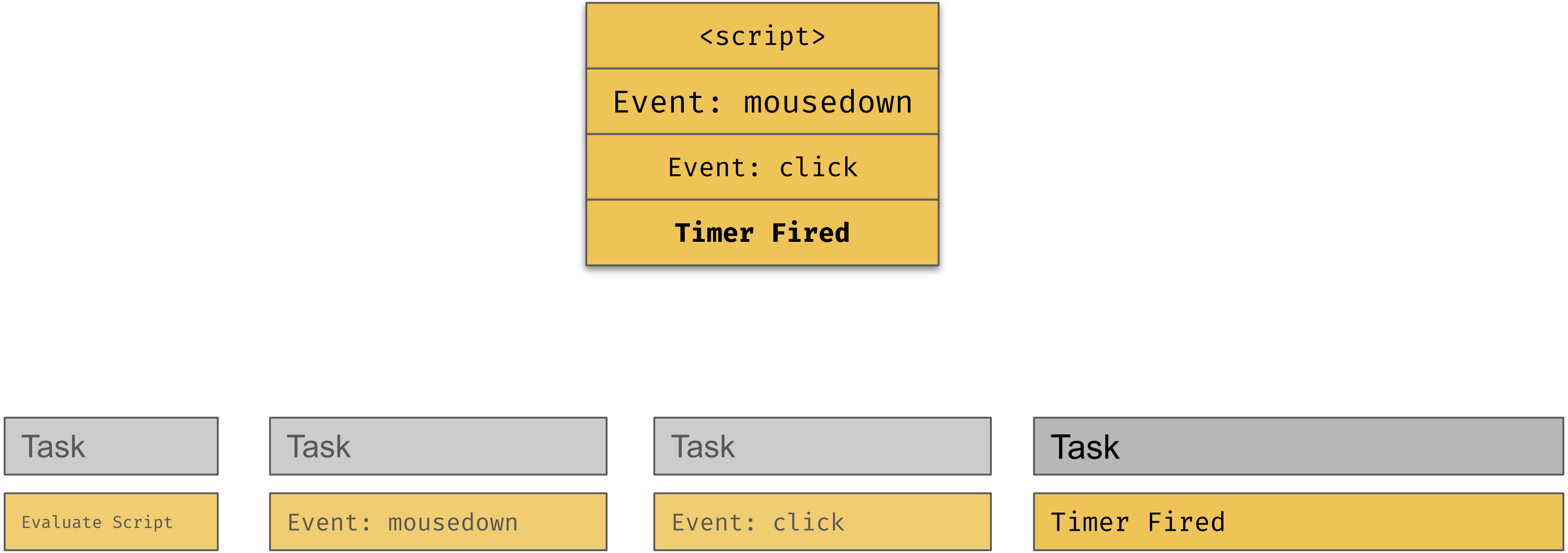
```
work();
work2();
```

Macrotask execution



```
doSmthOnClick();
```

Run-to-completion



<script>

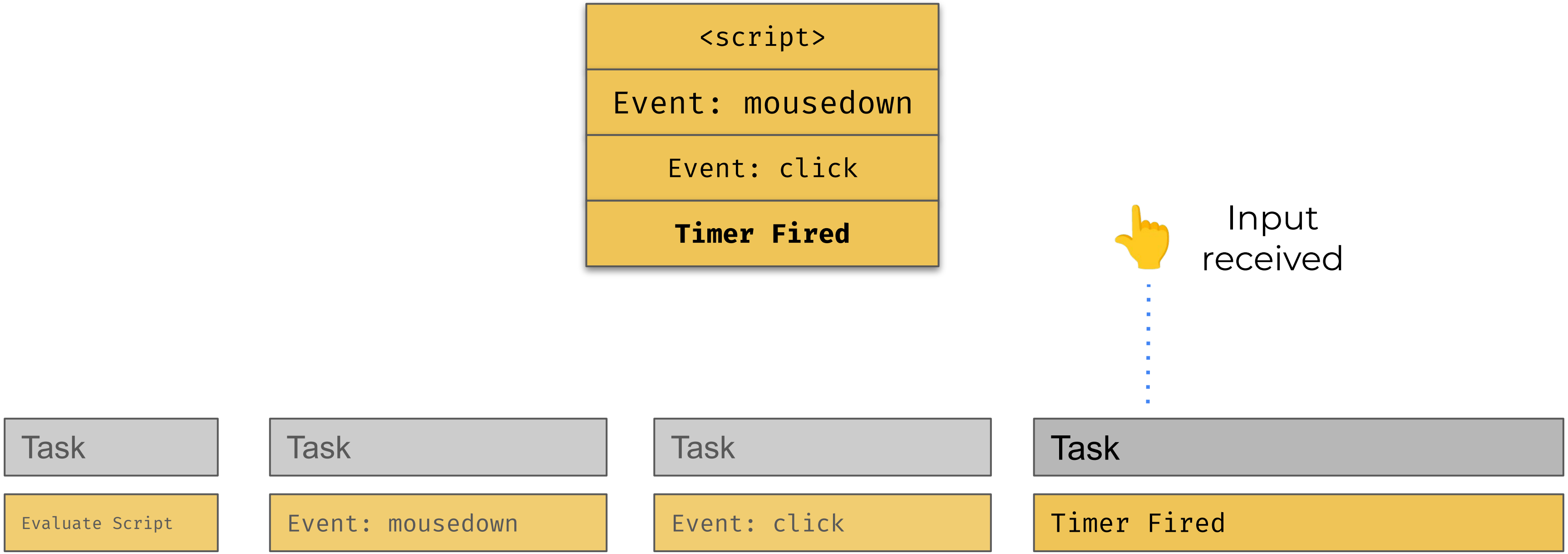
Event: mousedown

Event: click

Timer Fired

Task	Task	Task	Task
Evaluate Script	Event: mousedown	Event: click	Timer Fired

Run-to-completion



<script>

Event: mousedown

Event: click

Timer Fired

👆 Input received

Task

Evaluate Script

Task

Event: mousedown

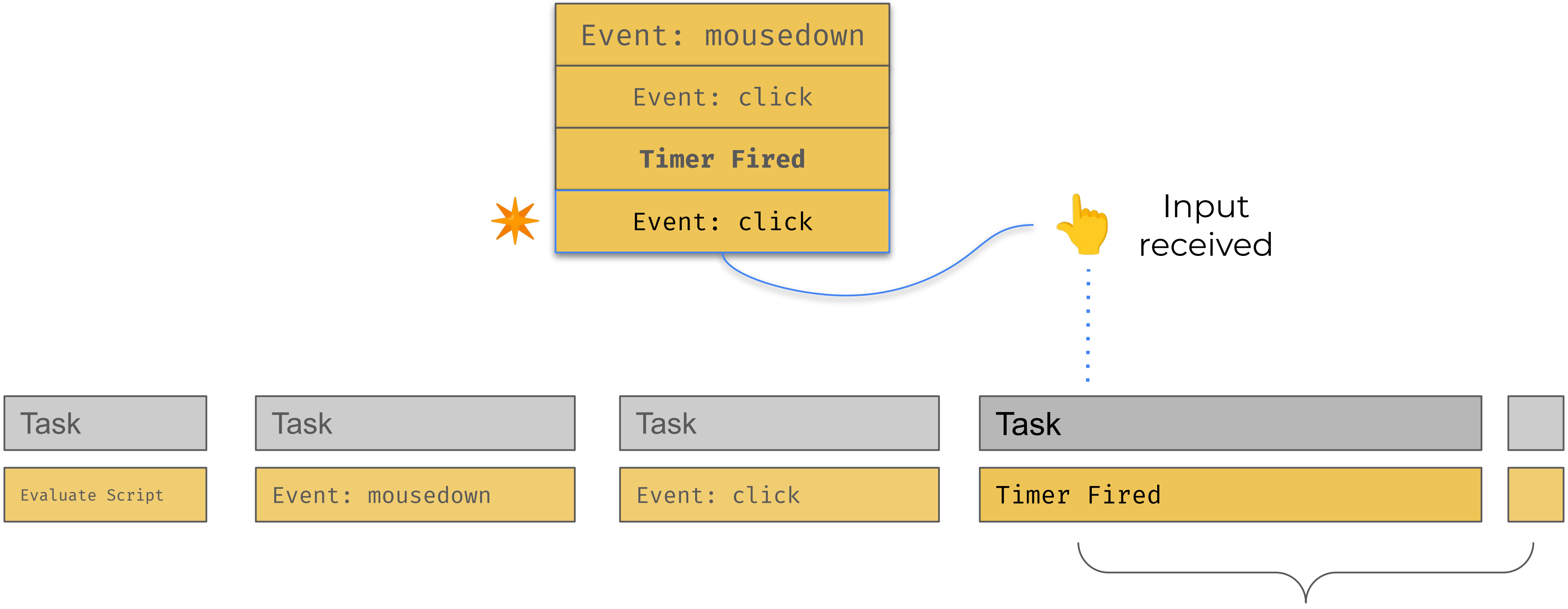
Task

Event: click

Task

Timer Fired

Run-to-completion



Event: mousedown

Event: click

Timer Fired

Event: click

Input received

Task

Evaluate Script

Task

Event: mousedown

Task

Event: click

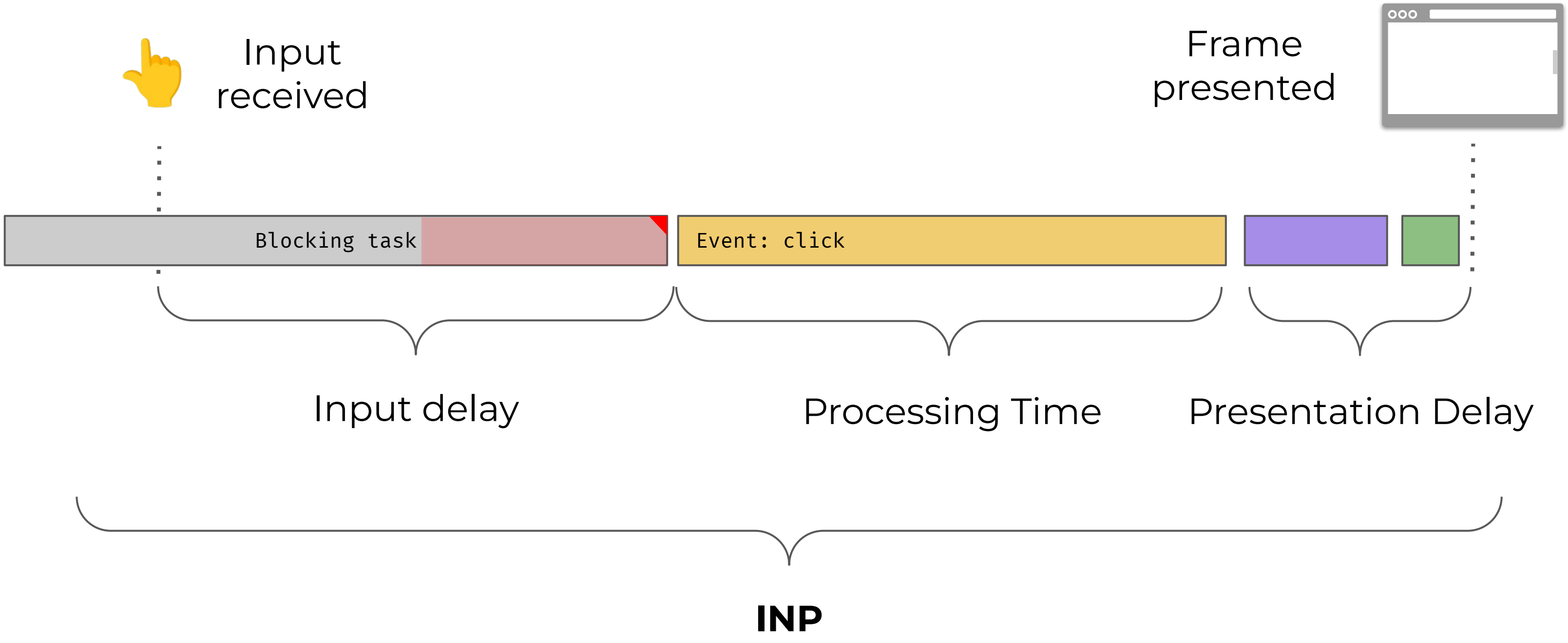
Task

Timer Fired

Input delay

"Run-to-completion"

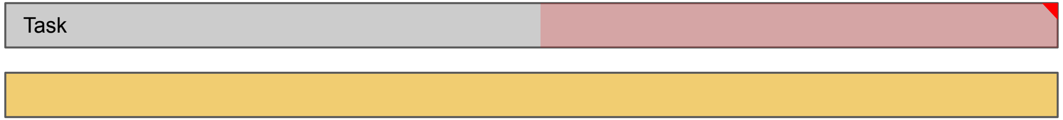
INP impacted by blocking task on the main thread

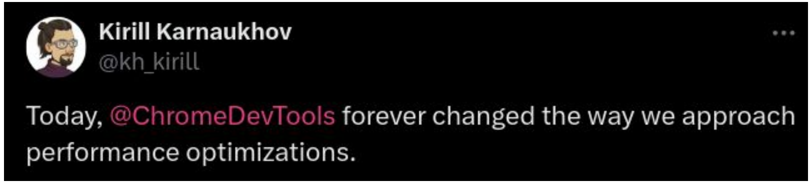


Live Demo

Run-to-completion

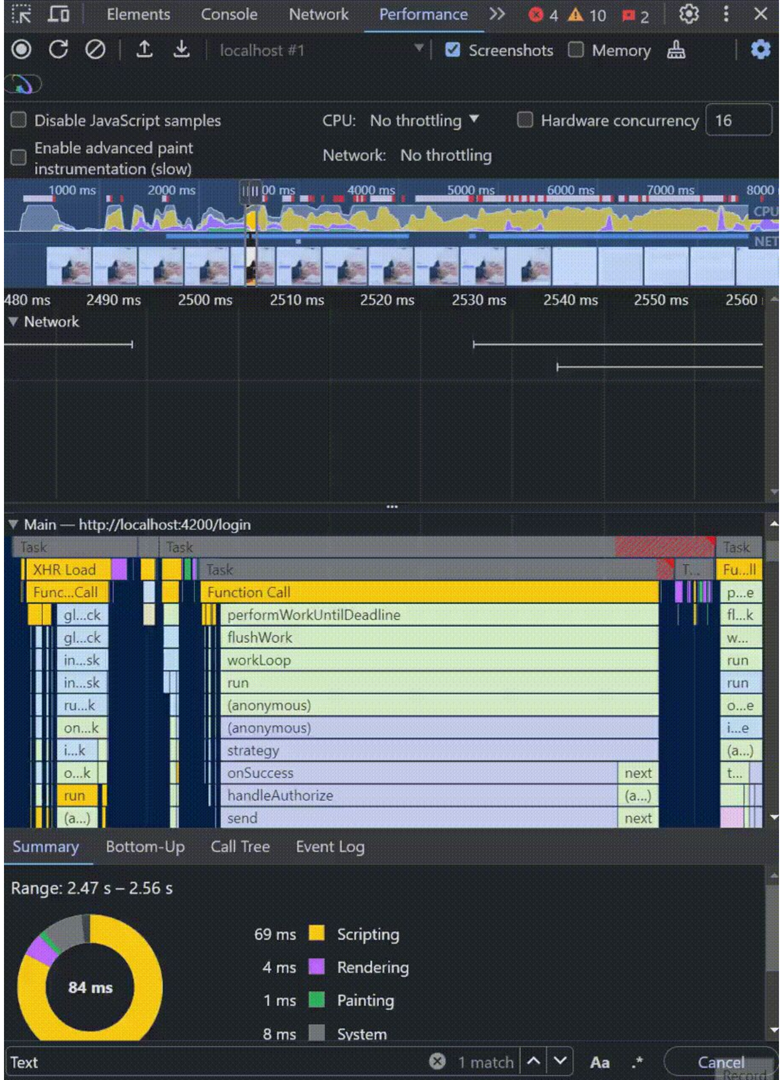
🤷 What to do with the long task then?





Kirill Karnaukhov
@kh_kirill

Today, @ChromeDevTools forever changed the way we approach performance optimizations.



□ Disable JavaScript samples

□ Enable advanced paint instrumentation (slow)

CPU: No throttling

□ Hardware concurrency 16

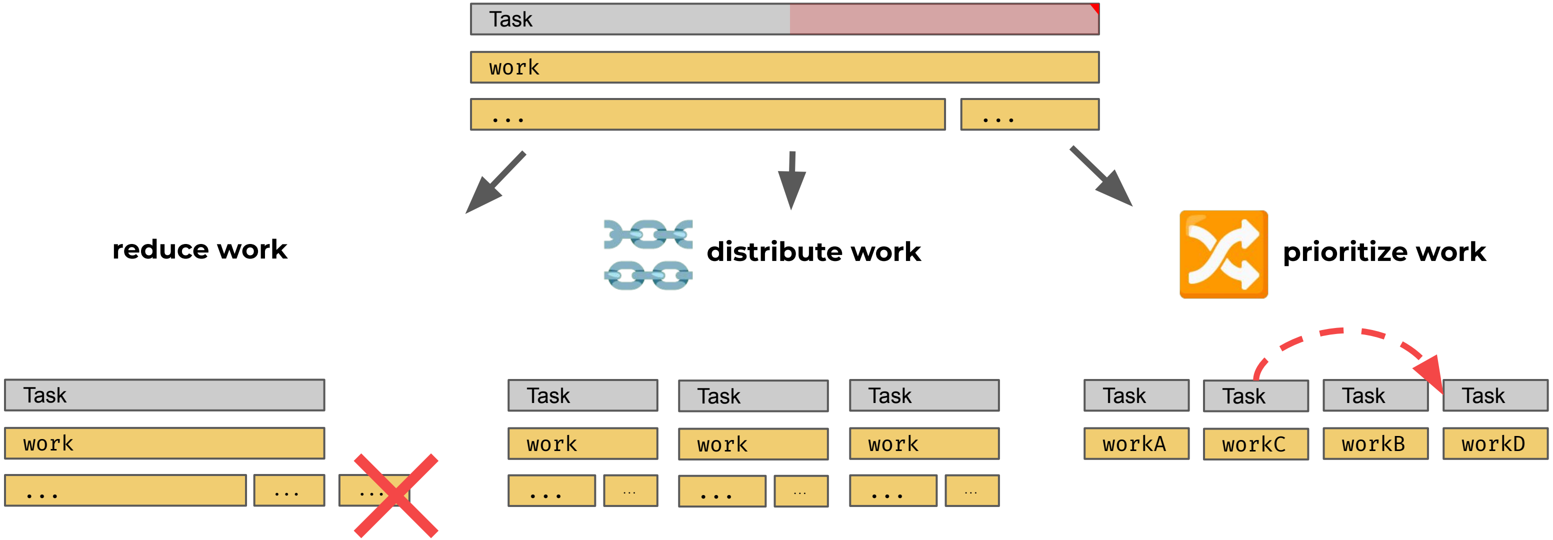
Network: No throttling

Main — http://localhost:4200/login

Range: 2.47 s – 2.56 s

69 ms	🟨	Scripting
4 ms	🟪	Rendering
1 ms	🟩	Painting
8 ms	⬜	System

3 ways to improve long tasks



preferred solution, but high effort + not always applicable

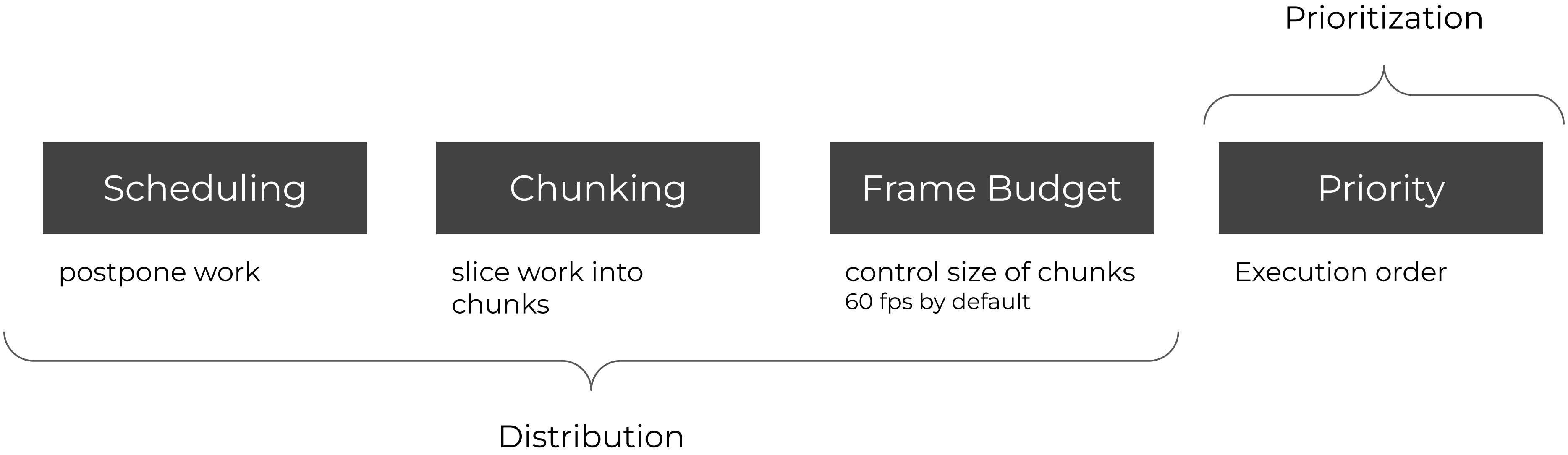
improve responsiveness by **splitting** apart the workload into **multiple tasks**

improve UX by **prioritizing** the important work to **display first**

RxAngular Concurrent Mode

🔗 rx-angular.io

Concurrent Mode: Conceptual Overview



Prioritization

Scheduling

postpone work

Chunking

slice work into chunks

Frame Budget

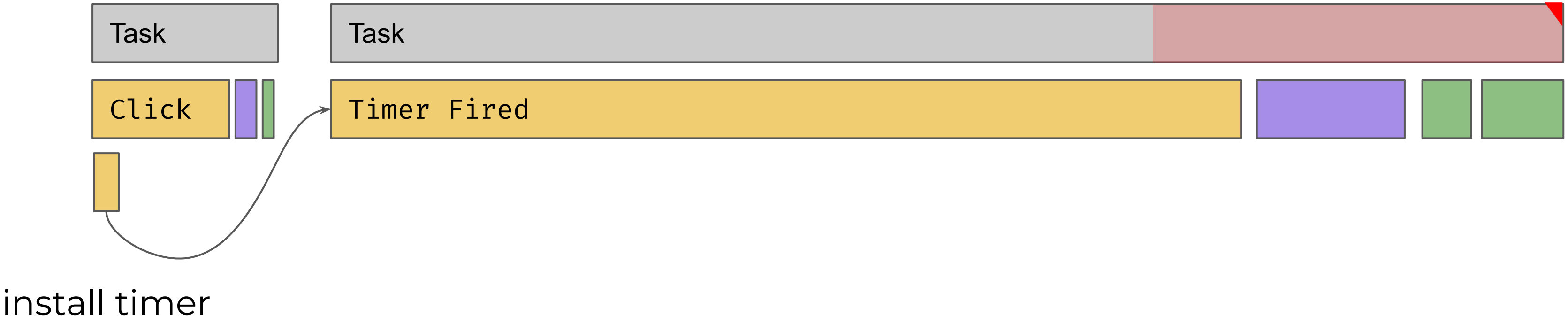
control size of chunks
60 fps by default

Priority

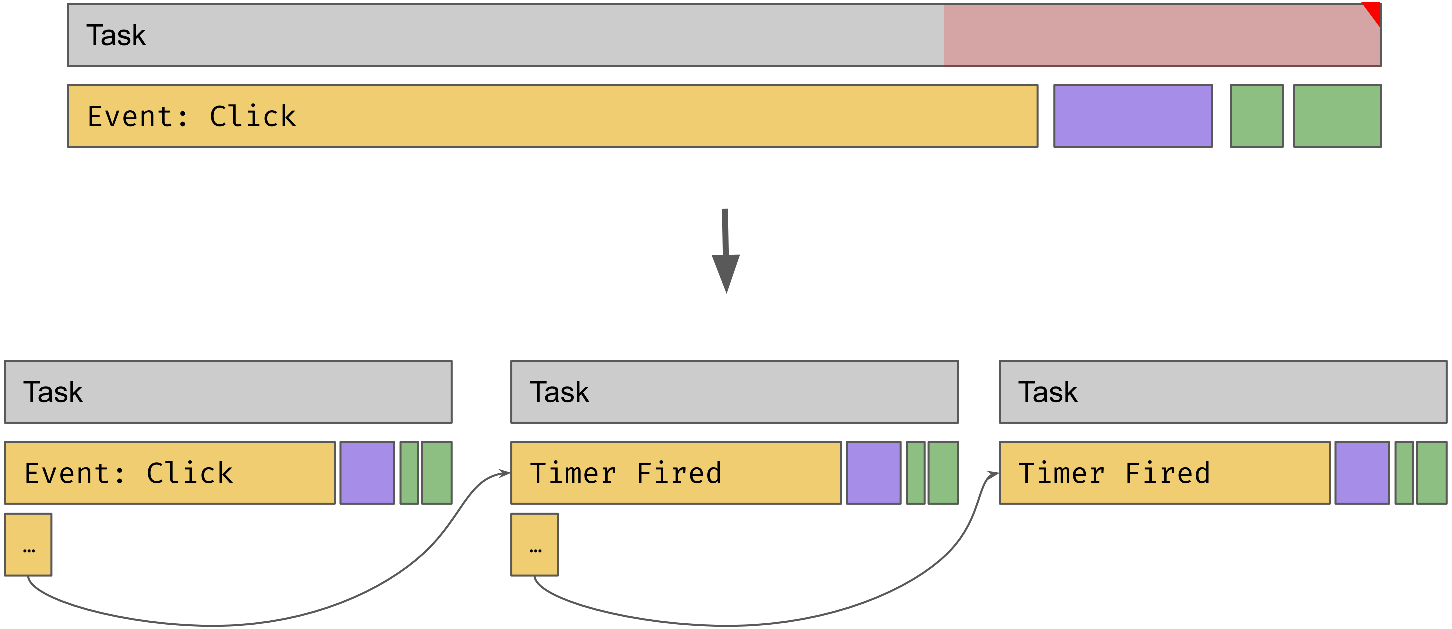
Execution order

Distribution

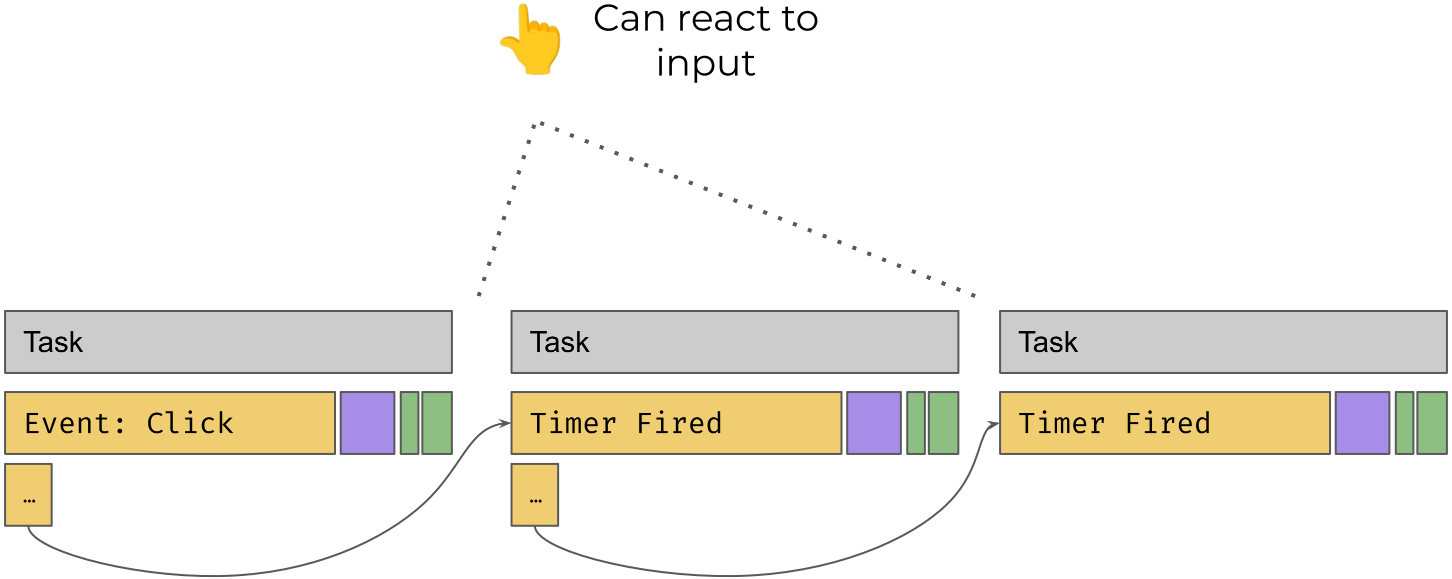
Concurrent Mode: Scheduling



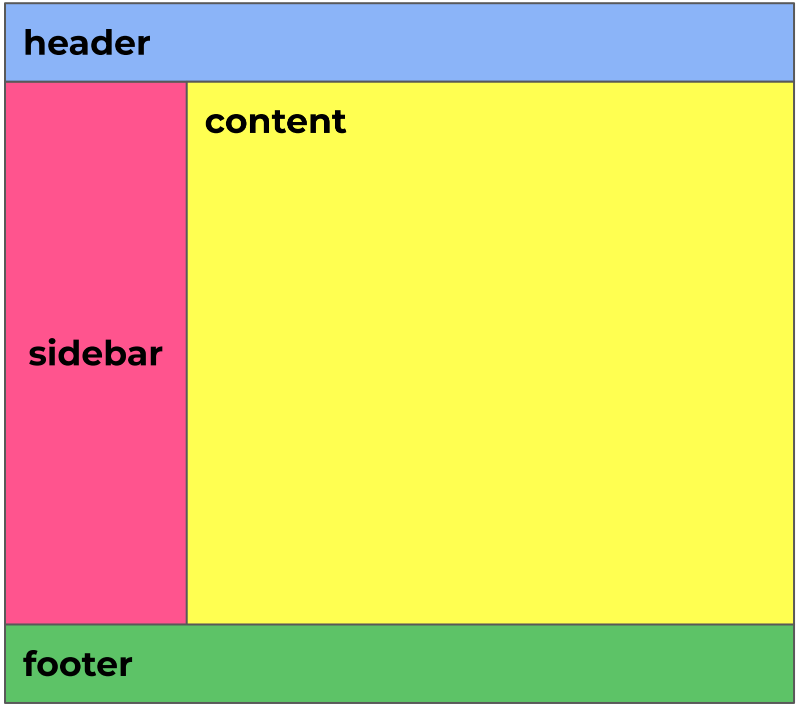
Concurrent Mode: Chunking



Concurrent Mode: Chunking



Chunk your template



header

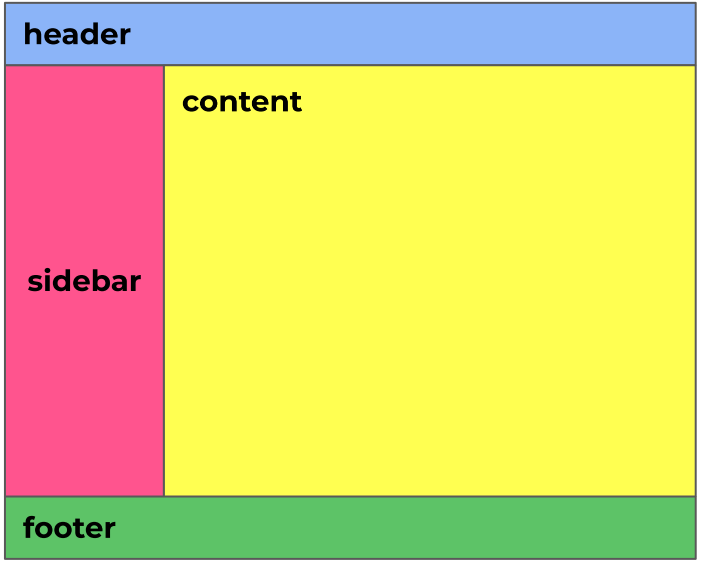
sidebar

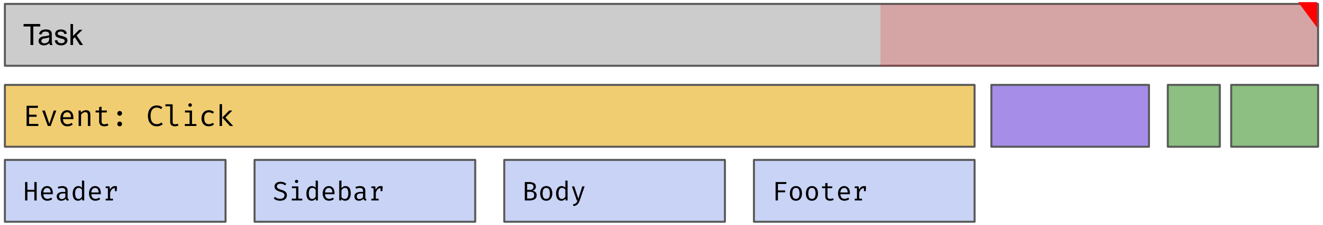
content

footer

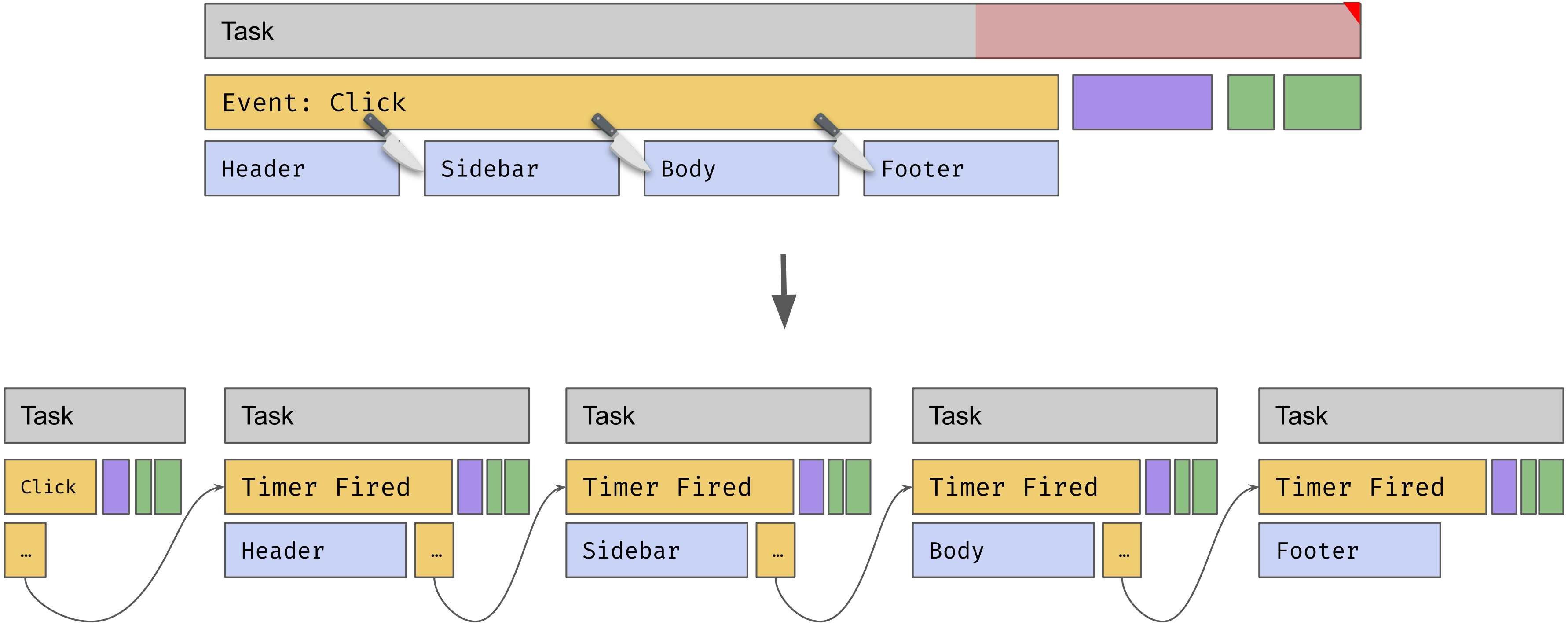
```
<!-- app-shell.component.html -->


<header />

<sidebar />

<body />

<footer />
```

Concurrent Mode: Chunking





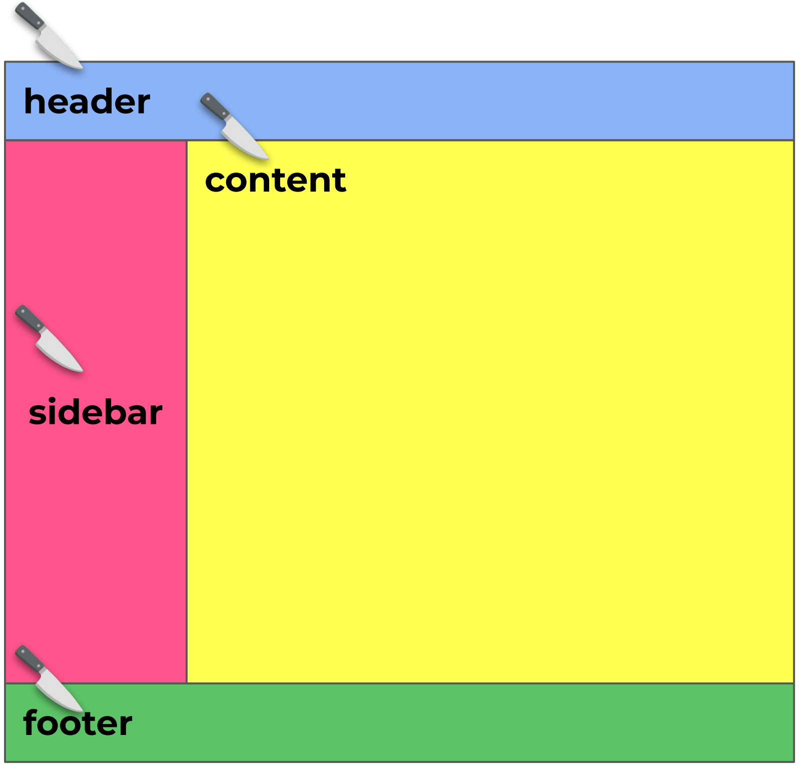
Concurrent Mode: Chunking



RxLet

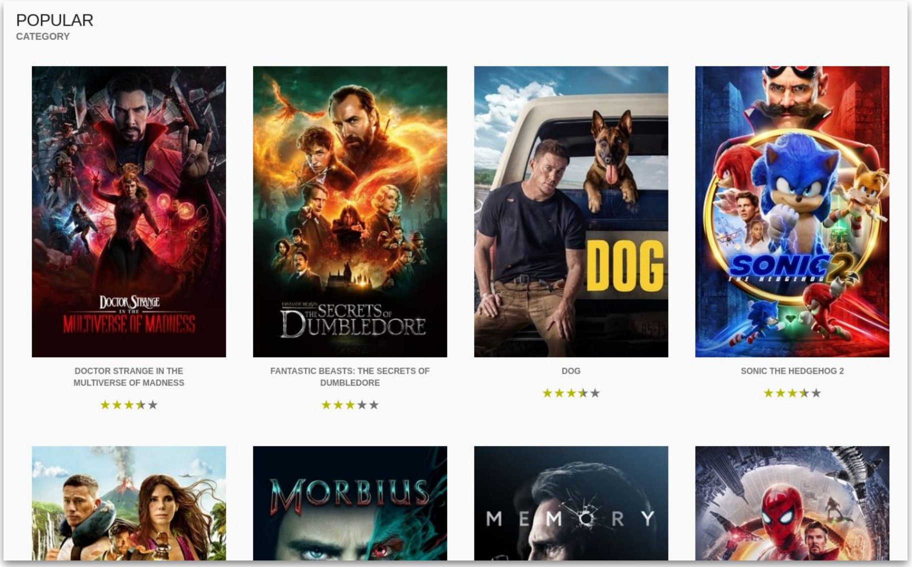
```
import { RxLet } from '@rx-angular/template';
```

Chunk your template



```
<!-- app-shell.component.html -->


<header *rxLet="[]" />
<sidebar *rxLet="[]" />
<body *rxLet="[]" />
<footer *rxLet="[]" />
```

Chunk your template: Lists



POPULAR
CATEGORY

DOCTOR STRANGE IN THE MULTIVERSE OF MADNESS
★★★★★

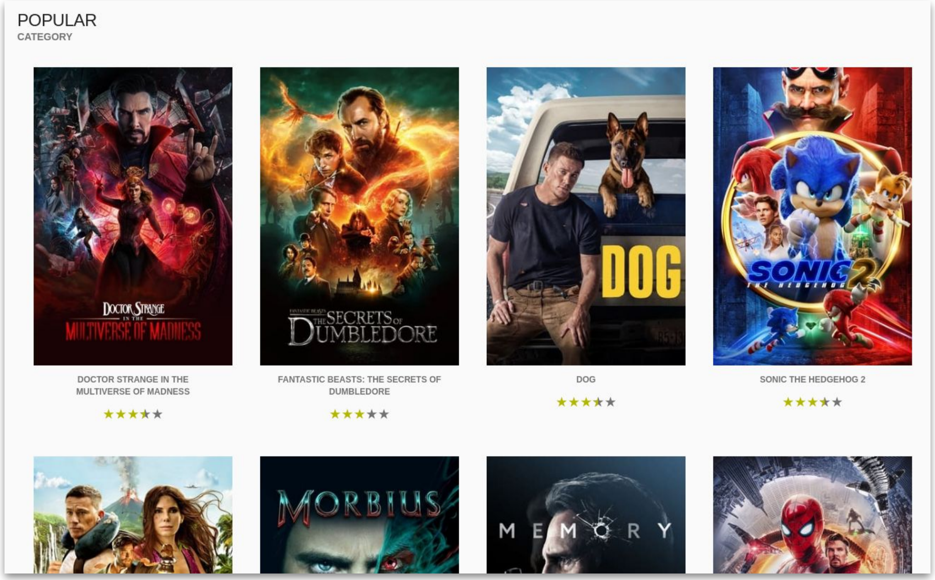
FANTASTIC BEASTS: THE SECRETS OF DUMBLEDORE
★★★★★

DOG
★★★★★

SONIC THE HEDGEHOG 2
★★★★★

```
<item
  *ngFor="let item of items"
  [data]="item" />
```

Chunk your template: Lists



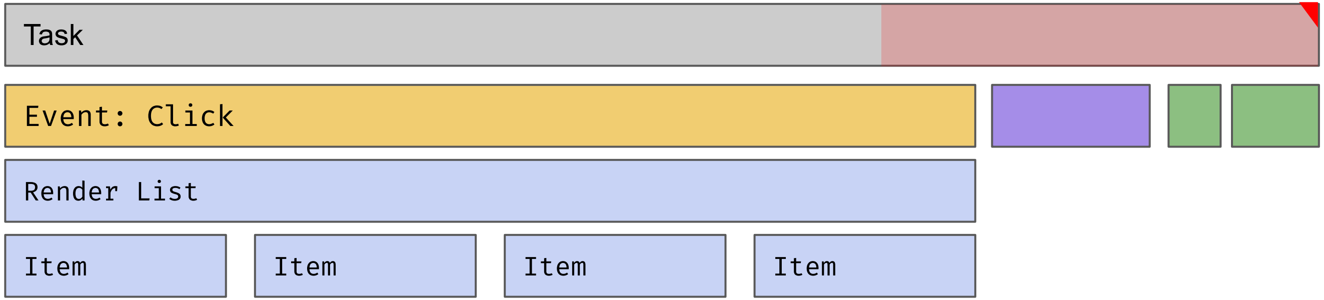
POPULAR
CATEGORY

DOCTOR STRANGE IN THE MULTIVERSE OF MADNESS
★★★★★

FANTASTIC BEASTS: THE SECRETS OF DUMBLEDORE
★★★★★

DOG
★★★★★

SONIC THE HEDGEHOG 2
★★★★★



Task

Event: Click

Render List

Item

Item

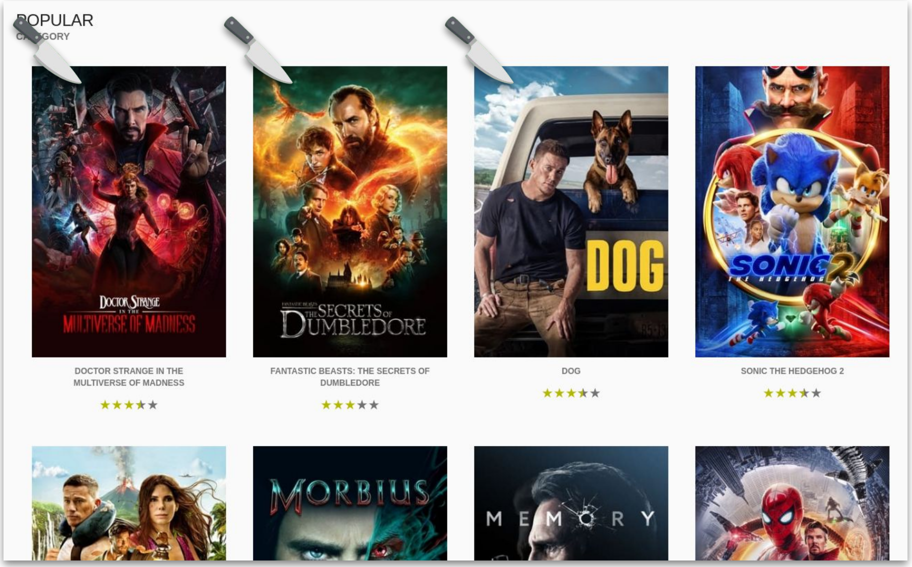
Item

Item

RxFor

```
import { RxFor } from '@rx-angular/template';
```

Chunk your template: Lists



POPULAR
CATEGORY

DOCTOR STRANGE IN THE MULTIVERSE OF MADNESS
★★★★★

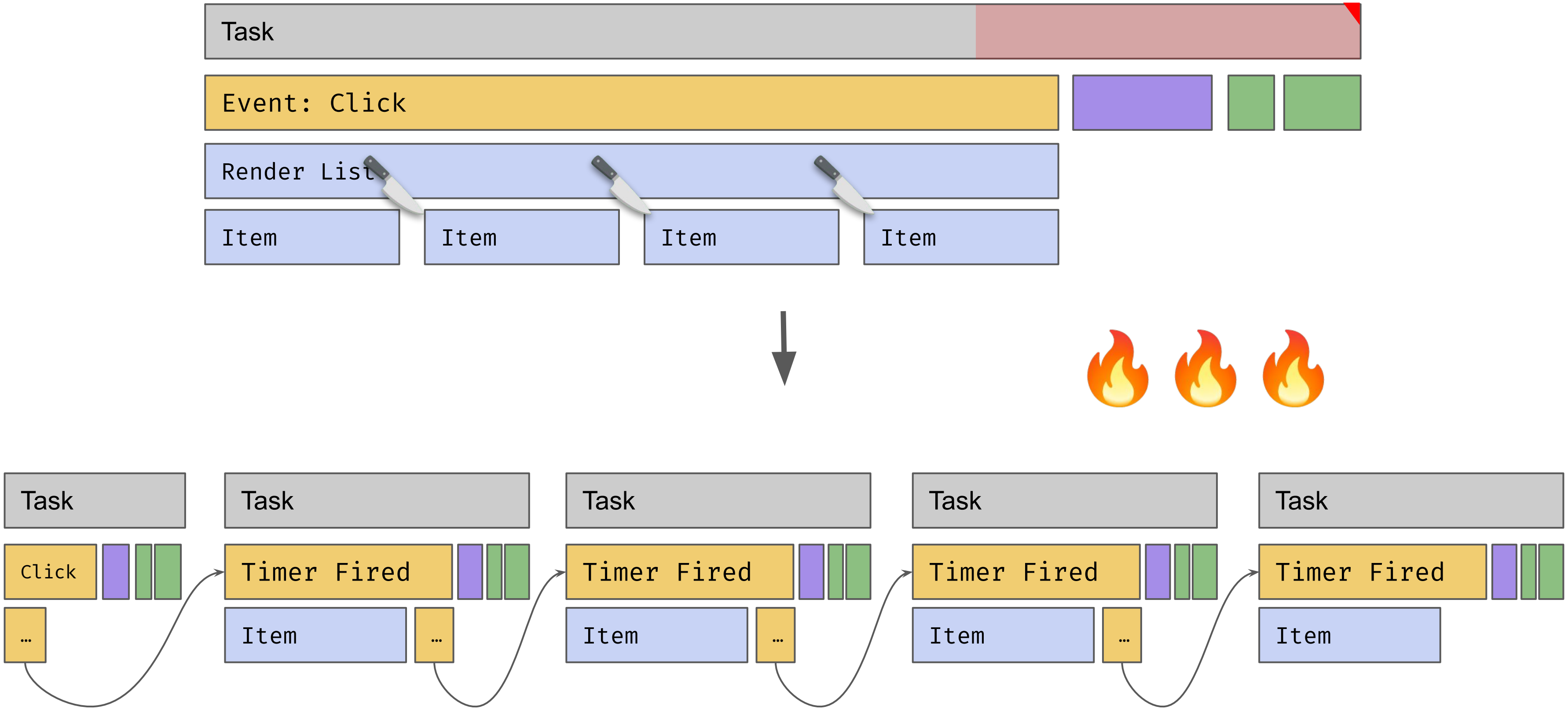
FANTASTIC BEASTS: THE SECRETS OF DUMBLEDORE
★★★★★

DOG
★★★★★

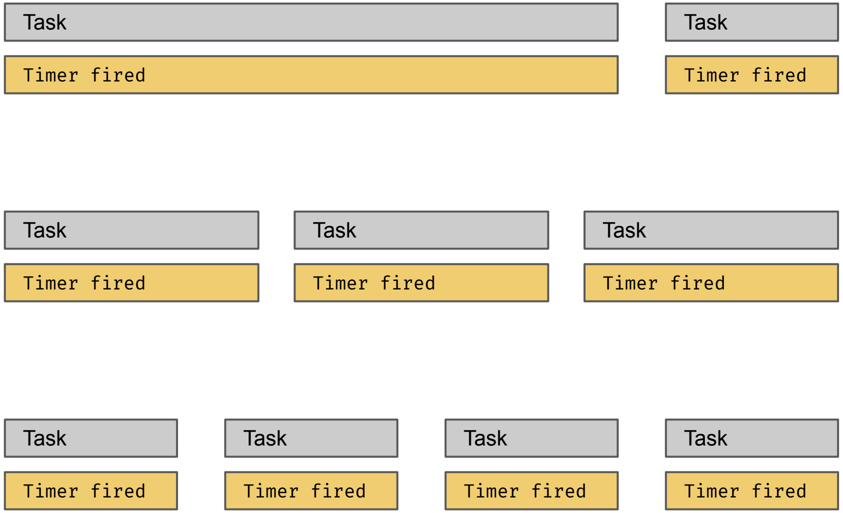
SONIC THE HEDGEHOG 2
★★★★★

```
<item
   *rxFor="let item of items"
   [data]="item" />
```

Chunk your template: Lists



Concurrent Mode: Frame budget



30 fps
~ 33.33ms each task

Task Task

Timer fired Timer fired

60 fps
~ 16.66ms each task

Task Task Task

Timer fired Timer fired Timer fired

90 fps
~ 11.11ms each task

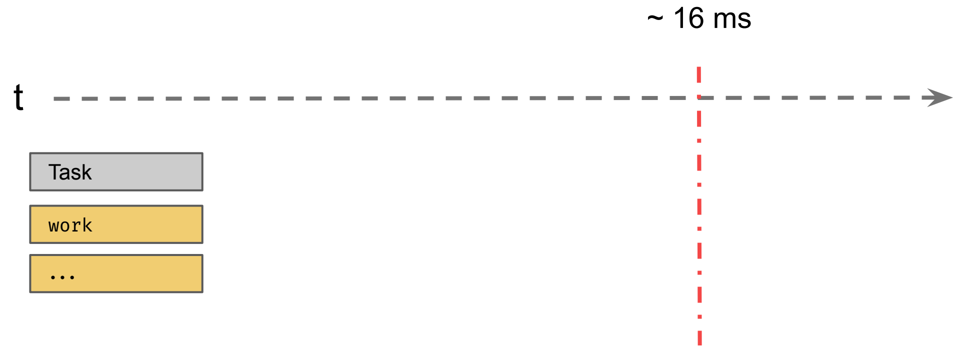
Task Task Task Task

Timer fired Timer fired Timer fired Timer fired

Concurrent Strategies: frame budget - chunk size computation

60 fps

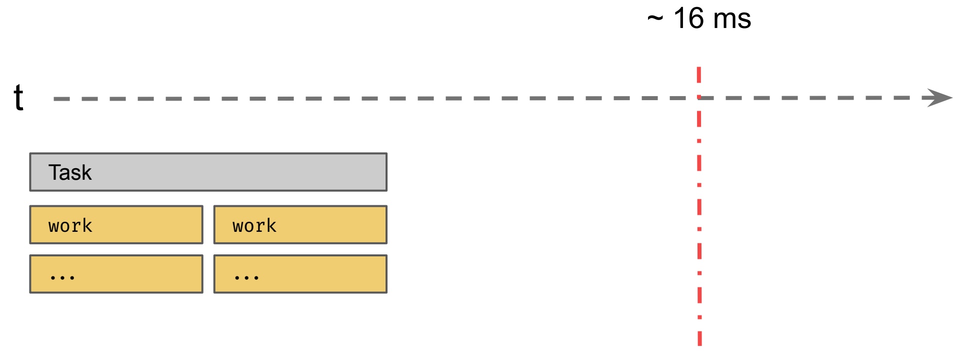
> 16.66ms each task



Concurrent Strategies: frame budget - chunk size computation

60 fps

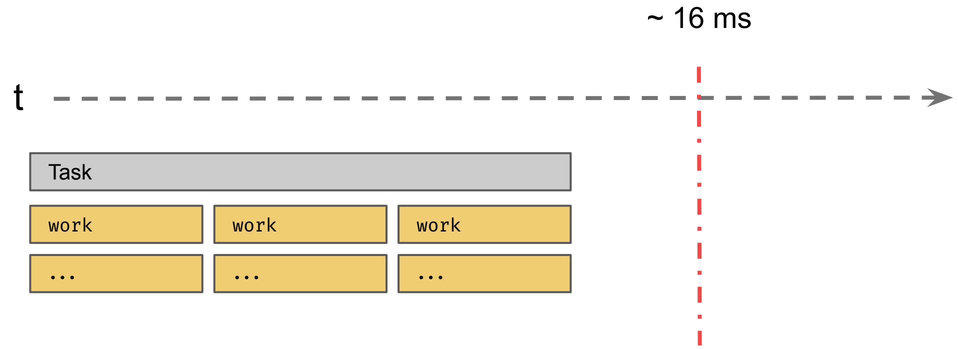
> 16.66ms each task



Concurrent Strategies: frame budget - chunk size computation

60 fps

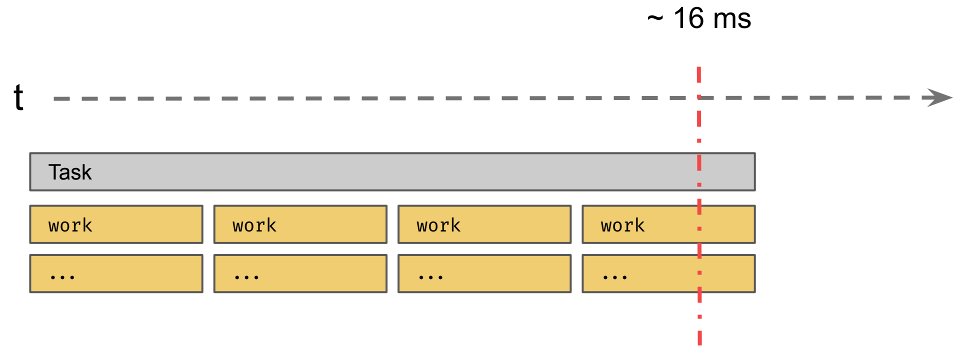
> 16.66ms each task



Concurrent Strategies: frame budget - chunk size computation

60 fps

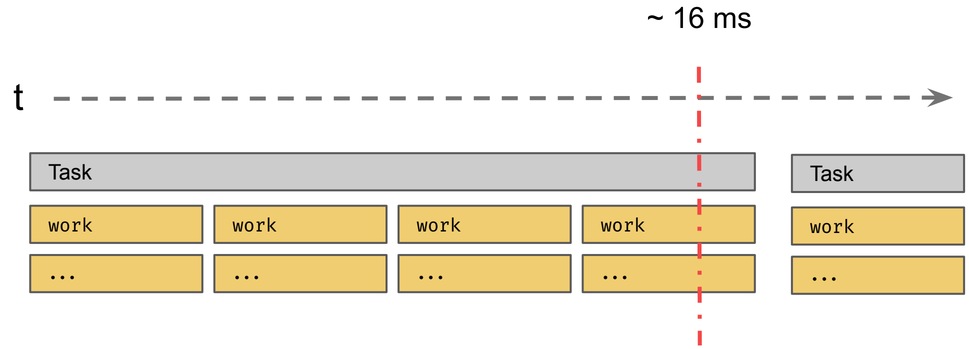
> 16.66ms each task



Concurrent Strategies: frame budget - chunk size computation

60 fps

> 16.66ms each task



~ 16 ms

t

Task

work

work

work

work

...

...

...

...

Task

work

...

Concurrent Strategies: Calculate Frame Budget

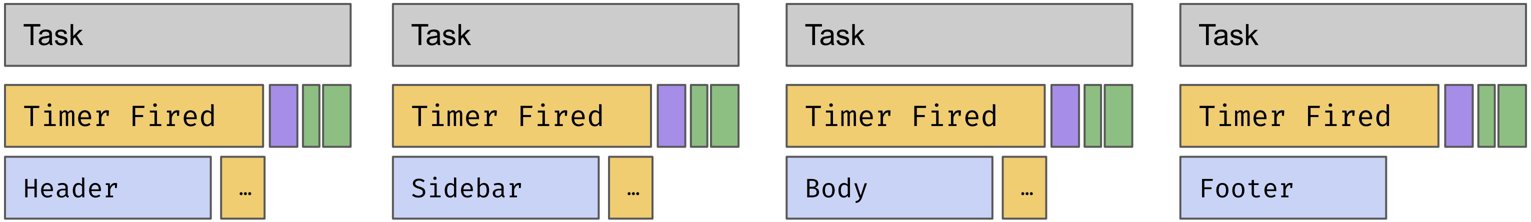
💡

```
while(task && runtime < threshold) {
  task = queue.pop();
  runtime += execute(task);
}
```

Live Demo

Distribute Template Work

Concurrent Mode: Priority



Task

Timer Fired

Header …

Task

Timer Fired

Sidebar …

Task

Timer Fired

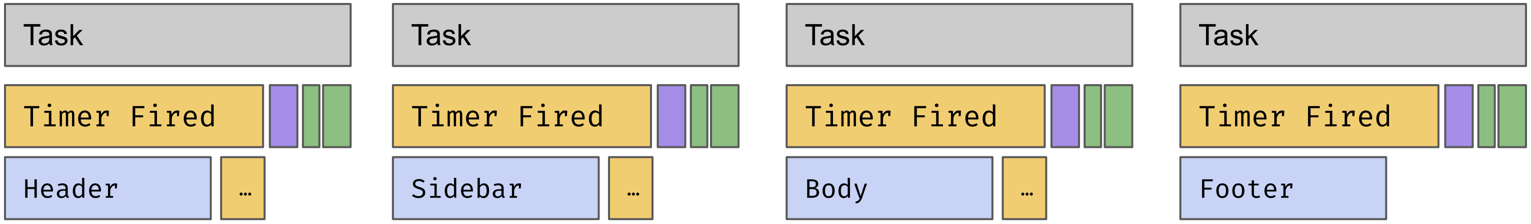
Body …

Task

Timer Fired

Footer

Concurrent Mode: Priority

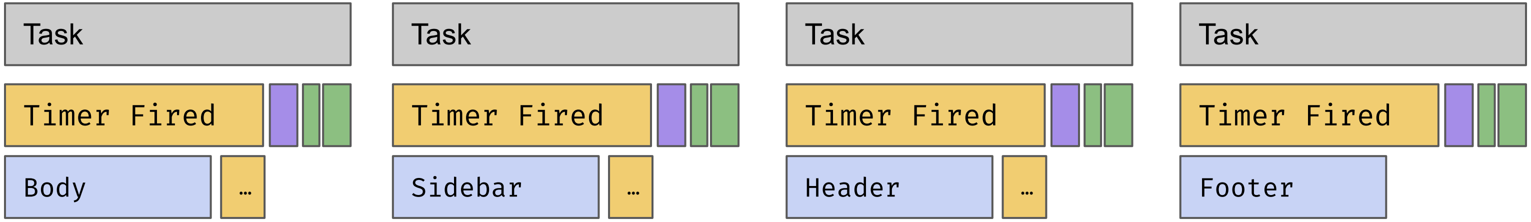


Task
Timer Fired
Header
…

Task
Timer Fired
Sidebar
…

Task
Timer Fired
Body
…

Task
Timer Fired
Footer



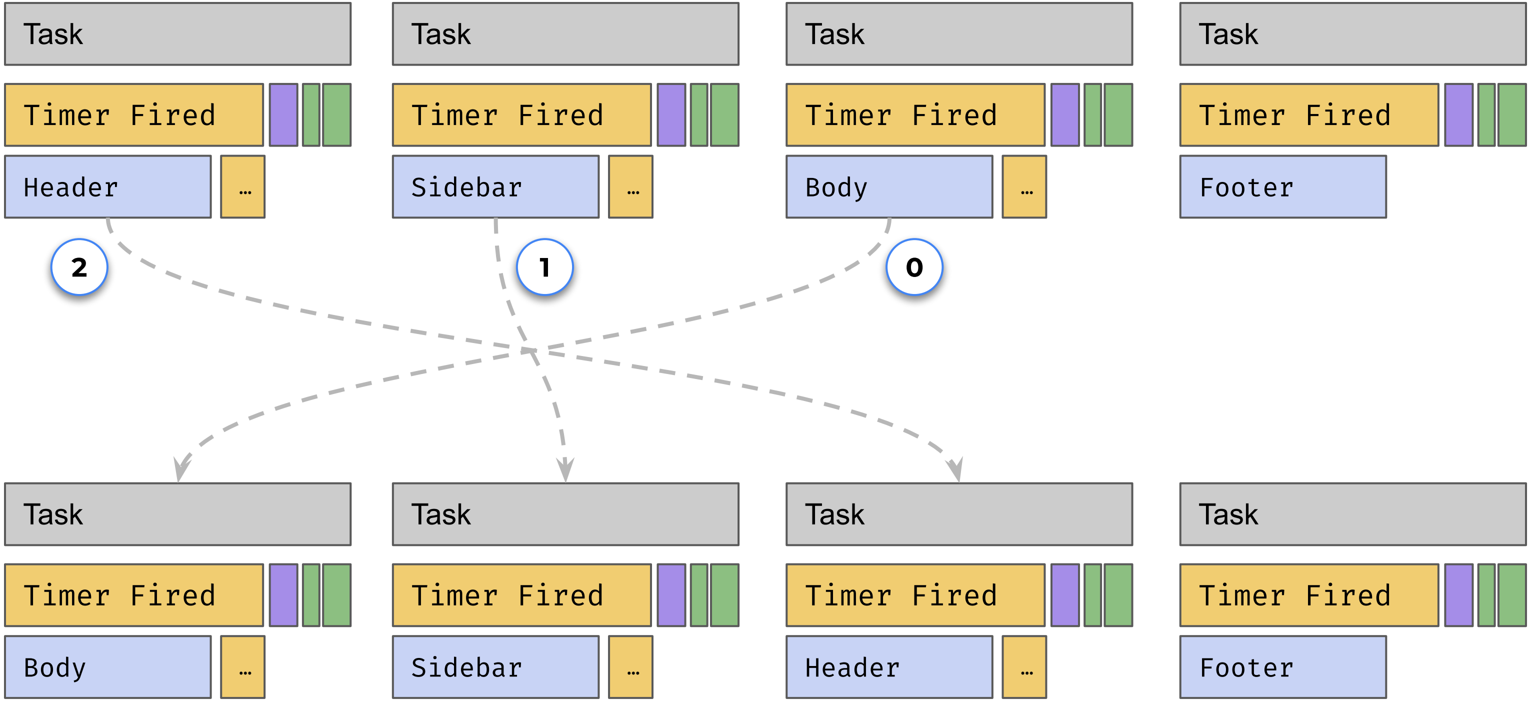
Task
Timer Fired
Body
…

Task
Timer Fired
Sidebar
…

Task
Timer Fired
Header
…

Task
Timer Fired
Footer

Concurrent Mode: Priority



Concurrent Mode: Priority

Immediate

Priority: 0

userBlocking

Priority: 1

normal

Priority: 2

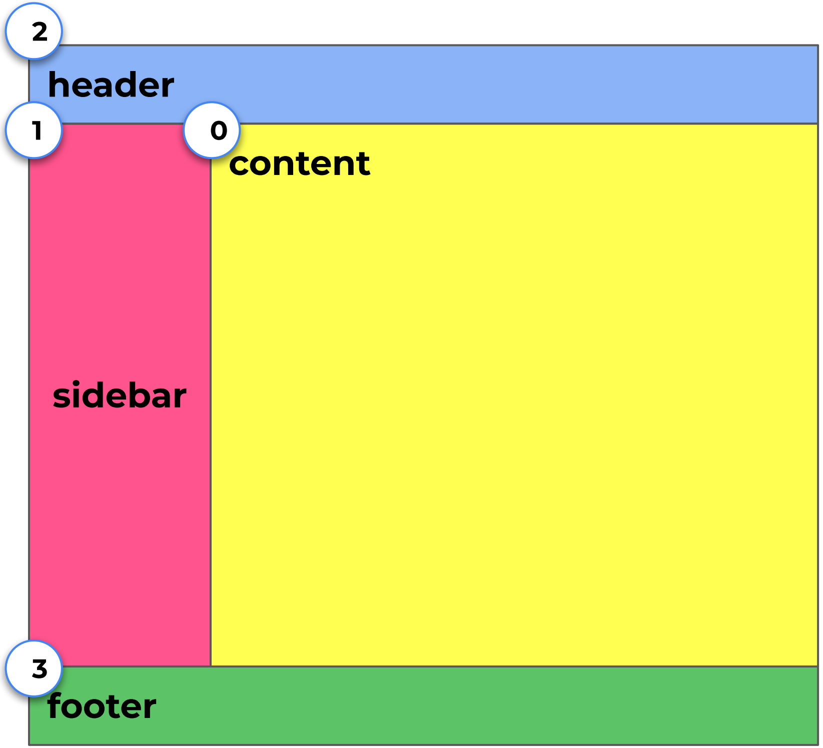
low

Priority: 3

idle

Priority: 4

Chunk your template



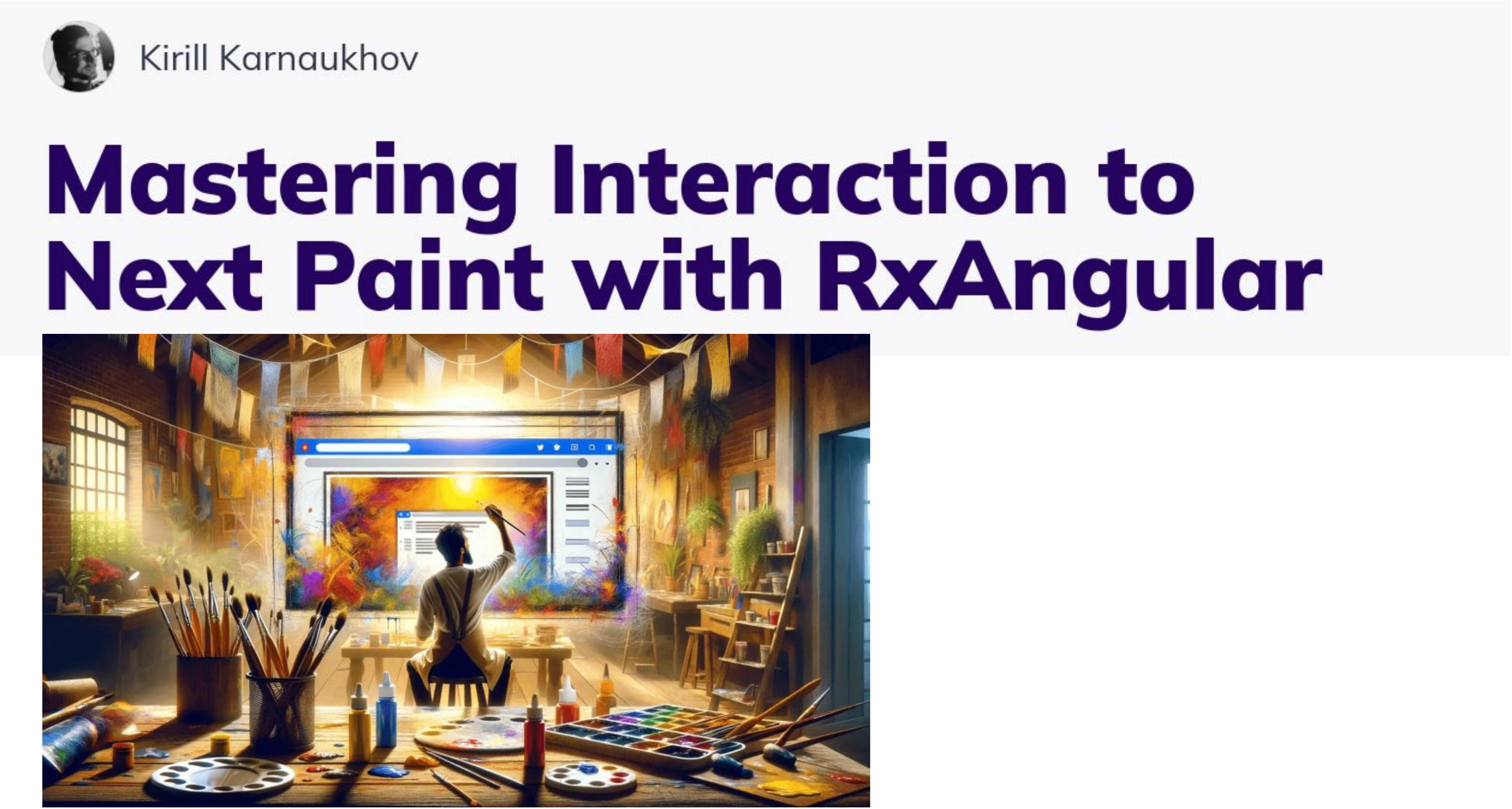
```
<!-- app-shell.component.html -->


<header *rxLet="[]; strategy: 'low'" />
<sidebar *rxLet="[];" />
<body *rxLet="[]; strategy: 'immediate'" />
<footer *rxLet="[]; strategy: 'low'" />
```

Live Demo

Prioritize Template Work

RxAngular
Performance & DX

check it out
&
⭐ us on github ⭐
https://github.com/rx-angular/rx-angular



Kirill Karnaukhov

Mastering Interaction to Next Paint with RxAngular

[Learn more about INP here](#)

Julian Jandl

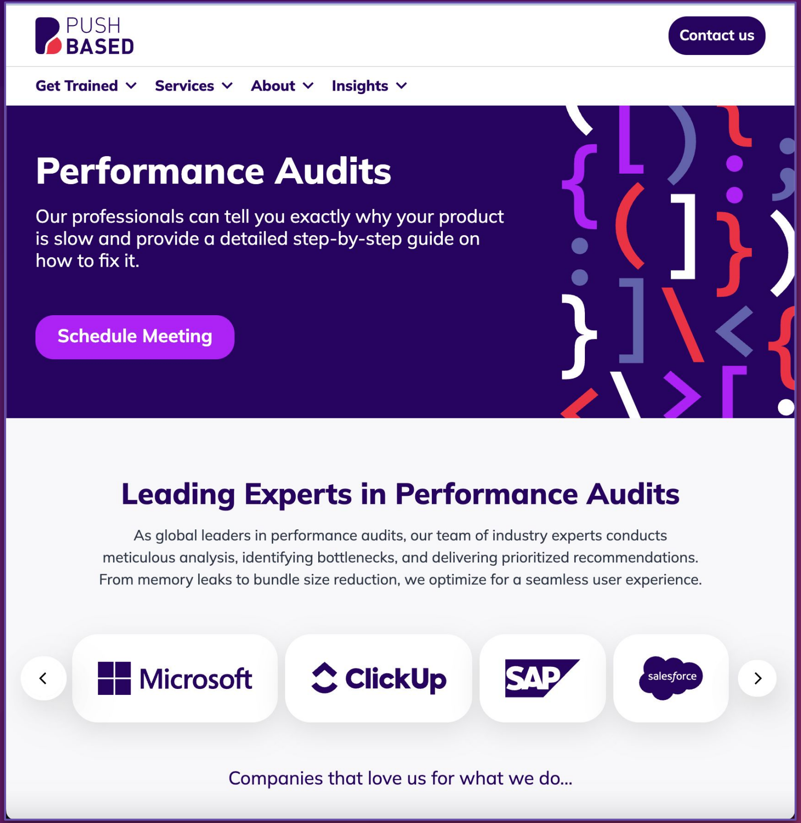
Performance Engineer

Trainer & Consultant
@rx-angular core maintainer

PUSH
BASED

@hoebbelsB

Web Performance Consulting



PUSH BASED

Contact us

Get Trained ⌄ Services ⌄ About ⌄ Insights ⌄

Performance Audits

Our professionals can tell you exactly why your product is slow and provide a detailed step-by-step guide on how to fix it.

Schedule Meeting

Leading Experts in Performance Audits

As global leaders in performance audits, our team of industry experts conducts meticulous analysis, identifying bottlenecks, and delivering prioritized recommendations. From memory leaks to bundle size reduction, we optimize for a seamless user experience.

‹ Microsoft ClickUp SAP salesforce ›

Companies that love us for what we do...

Grazie per l'attenzione!

Slides

Repo