Smarter Change Detection Techniques

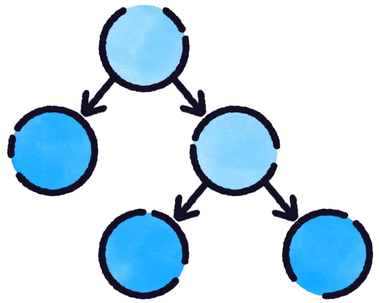
Brian Treese
Chief of User Experience

www.youtube.com/@briantreese | www.socreate.it

Does It Still Matter?

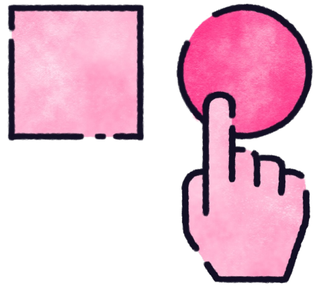
Yes!

OnPush With Zones



Input changes

Manual triggers



Certain events

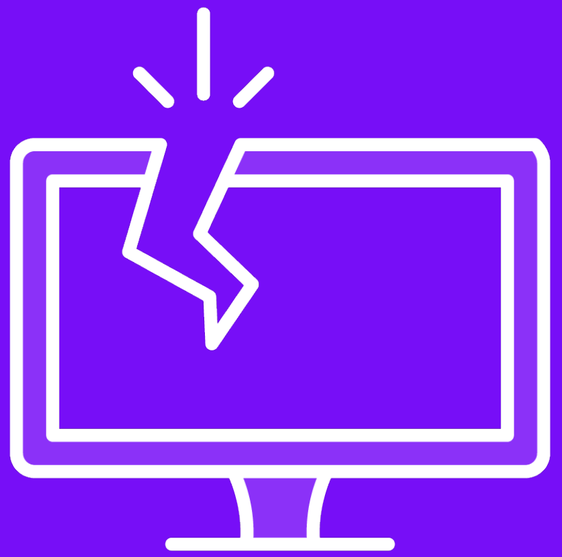
Why It Still Matters

You control when

You control where

Only checks what actually changed

We Still Need Boundaries!



What to we do when things break?

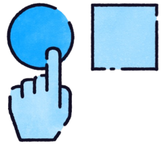
Manual Change Detection: Fine-tuned Control

Manual Change Detection

Not the
first choice

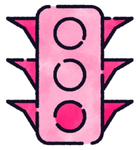
But still an
essential tool

What Triggers Change Detection Now?

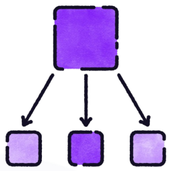


Event bindings

Manual change detection

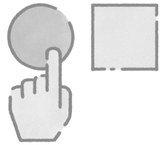


Signal changes



Calling setInput()

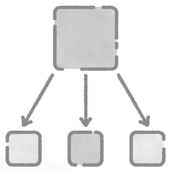
What Triggers Change Detection Now?



Event bindings

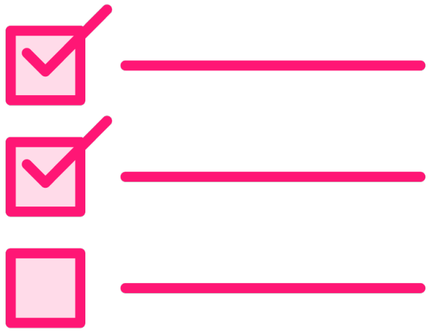
Manual change detection

Signal changes



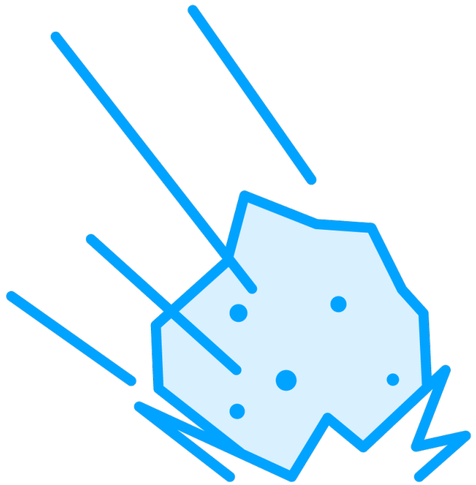
Calling setInput()

ChangeDetectorRef



markForCheck()

Self, ancestors, and children
Updates on the next cycle



detectChanges()

Self and children only
Updates immediately

Key Concepts

markForCheck: batch for next cycle (bubbles up)

detectChanges: update now (subtree only)

Needed less often with signals

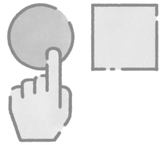
Async Pipe with Observables: Still a Smart Move?

The Async Pipe

No subscriptions needed!

Still a valid tool

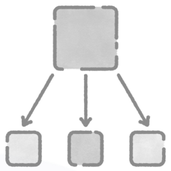
What Triggers Change Detection Now?



Event bindings

Manual change detection

Signal changes



Calling setInput()

Yes!

It still works

Is It Still Useful?

Yes, it still works great!

Less critical with signals

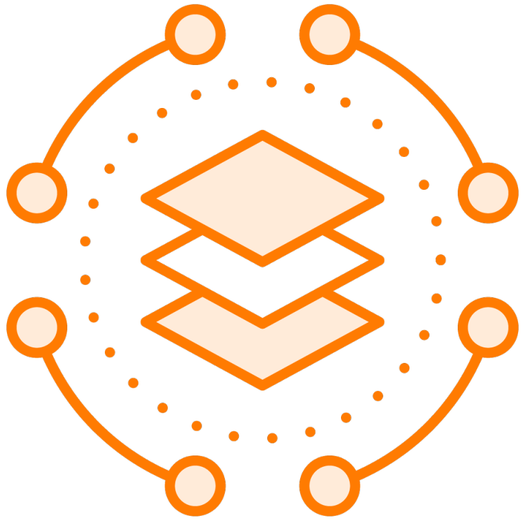
Signals!

Signals: Angular’s New Reactive Power Tool

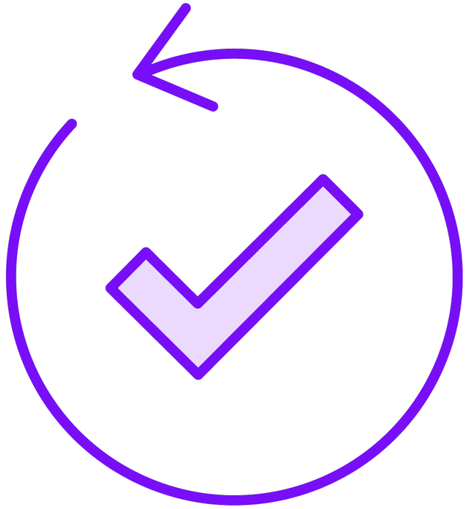
What If?

Signals!

Signals?

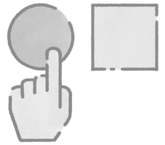


A reactive state primitive



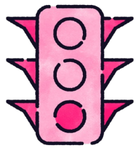
Automatically update the UI when values change

What Triggers Change Detection Now?

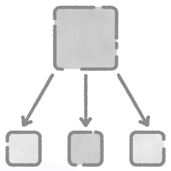


Event bindings

Manual change detection



Signal changes



Calling setInput()

Key Concepts

Use signals as much as possible

Reactive, targeted updates

Cleaner code

Better performance

From Observables to Signals: The Modern Path

Already Have Observables?

toSignal()

What Is toSignal()?

Converts observables to signals

Replaces the async pipe... sort of

Keeps RxJS logic while embracing signals

Key Concepts

No manual subscriptions

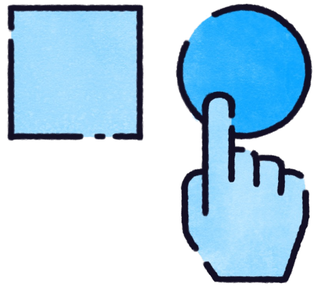
No need for async pipe

Provides a migration path

Improves performance

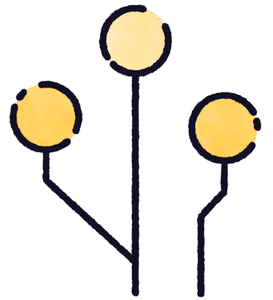
Fixing Change Detection for Dynamic Components

What We've Seen so Far

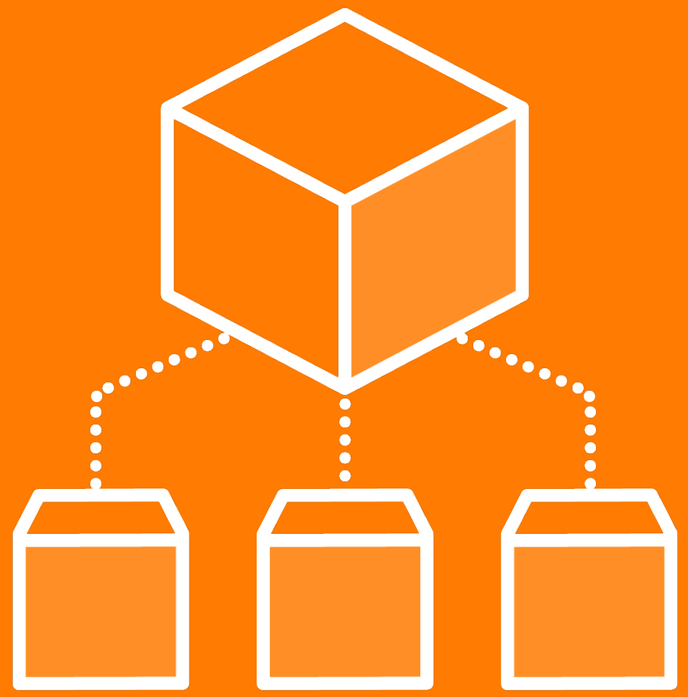


Certain events

Signals

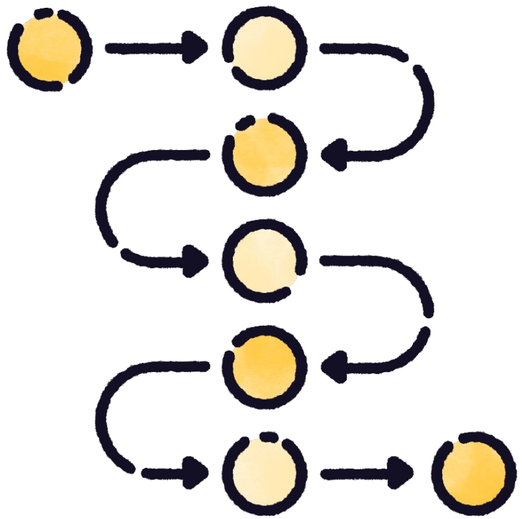


Async Pipe



Dynamic Components!

Dynamic Components and Change Detection

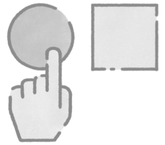


Need to update properly

Old patterns don’t work!

Need a modern approach

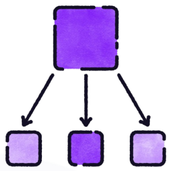
What Triggers Change Detection Now?



Event bindings

Manual change detection

Signal changes



Calling setInput()

Key Concepts

Use setInput() if you can!

Just works with Zoneless

It’s just better!

Real-world Examples