Angular’s Change Detection Mechanism

Change Detection

- Change detection is the process of capturing the internal state of a program and its changes, and projecting it to the user
- Change detection can be triggered by async tasks:
  - Events (Action on the dom, or custom events)
  - XHR (Http requests to fetch data)
  - Timers (timeouts and intervals)
- Angular uses Zones to implement change detection
- Angular comes with its own zone called NgZone
- Each component has its own change detector
- Angular exposes an API for CD called ChangeDetectorRef

A little more on Zones

- A Zone is an execution context that persists across async tasks

```
Zone.current.fork({}).run(function () {
    Zone.current.inTheZone = true;

    setTimeout(function () {
        console.log('in the zone: ' + !!Zone.current.inTheZone);
    }, 0);
});
```

- Zone monkey-patches all methods which cause async tasks to run in a zone

```
function zoneAwareAddEventListener() {...}
function zoneAwareRemoveEventListener() {...}
function zoneAwarePromise() {...}
function patchTimeout() {...}
window.prototype.addEventListener = zoneAwareAddEventListener;
window.prototype.removeEventListener = zoneAwareRemoveEventListener;
window.prototype.promise = zoneAwarePromise;
window.prototype.setTimeout = patchTimeout;
```

Continued...

- Zones can be created, forked, and extended
- The forked zone contains the methods:
  - onZoneCreated – Runs when zone is forked
  - beforeTask – Runs before a function called with zone.run is executed
  - afterTask – Runs after a function in the zone runs
  - onError – Runs when a function passed to zone.run will throw an error
- We can extend the Zone to include methods we need.

NgZone

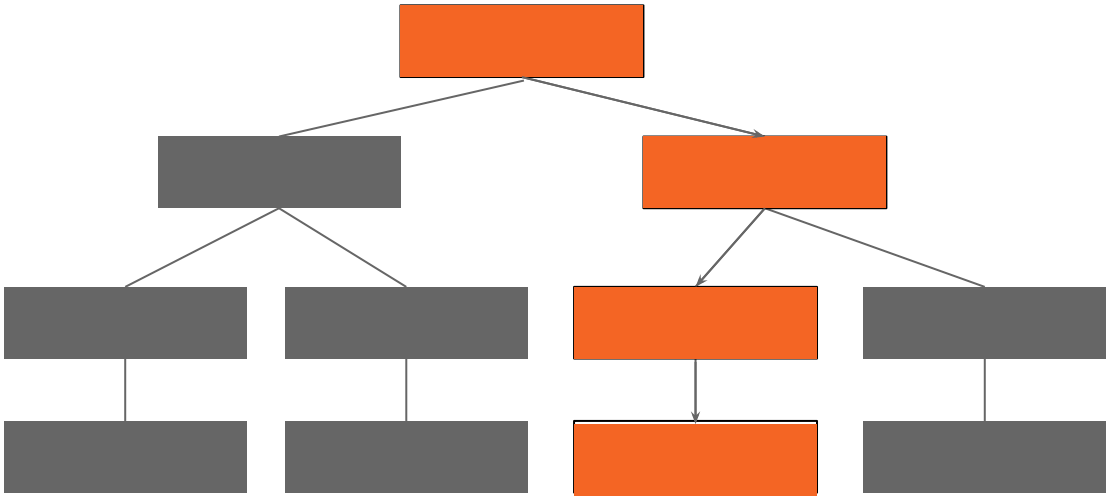
- NgZone is a forked zone that extends the zone API
- NgZone can be imported from @angular/core
- NgZone adds the following custom events that we can subscribe to:
  - onUnstable() - Notifies when code entered Angular Zone
  - onMicrotaskEmpty() - Notifies when there are no more microtasks
  - onStable() - Notifies when the last onMicrotaskEmpty has run
  - onError() - Notifies that an error has occurred
- It contains a method runOutsideAngular() to execute code outside Angular's zone which will not trigger change detection.
- You can re-enter the Angular zone by invoking the NgZone.run() method

Change Detection in Ng

- Since Angular uses Zones, change detection is triggered through monkey-patched native methods and does not require anything more
- Angular internally contains an ApplicationRef which controls CD
- Whenever an onMicrotaskEmpty() event is fired, Angular executes a tick() function which initiates change detection for all change detectors.
- Each component has its own change detector.
- Change detection is performed top to bottom, starting from the root component and flowing down the change detector tree
- By default, Angular detects changes for all components when event is fired
- We can use ChangeDetectionStrategy to prevent CD where required

Change Detector Ref

- ChangeDetectorRef is responsible for performing change detection
- It consists of the methods:
  - markForCheck()
  - detach()
  - detectChanges()
  - checkNoChanges()
  - reattach()
- We can import ChangeDetectorRef() from @angular/core
- Can be used to force change detection
- Can be used to exclude components from change detection
- Can be used to override ChangeDetectionStrategy.


AJAX / DOM Event
/ setTimeout
Perform Change Detection as usual
Mark path until root for check

@Directive and Angular Directives

Next Section

Credits

Special thanks to all the people who made and released these awesome resources for free:

- Presentation template by SlidesCarnival
- Photographs by Unsplash