AngularJS performance Tips

- AngularJS is the most popular JavaScript framework as of now - it’s being used for creating websites, and mobile apps (Ionic Framework).

- Even though it is most popular, It’s not as fast as it should be.

- Here are a few small tips on making your code faster, and more memory efficient.

Use bind once when possible

- A new feature landing in AngularJS 1.3 is the ability to bind an expression to a view without adding a watcher and constantly digesting it.

- Bind once aims to go some of the way towards alleviating this by allowing you to decrease the number of expressions being watched, making the digest loop faster and your application more performant.

Debouncing your ng-model updates

- One of the biggest bottlenecks in Angular 1.x is the `$digest` cycle.
- For example if you have a text field using an `ng-model`, whenever you type in a value into this field, it will trigger a `$digest` cycle causing Angular to update all watchers and bindings in the app to see if anything has changed.
- In version 1.3 of Angular, a native debounce feature was introduced which can be supplied to `ngModelOptions`.

Use ng-if instead of ng-show

- `ng-if` removes or recreates a portion of the DOM tree based on an expression.

- When an element is removed using `ng-if` its scope is destroyed and a new scope is created when the element is restored.

- The `ng-show` directive shows or hides the given HTML element based on the expression provided to the `ng-show` attribute. `ng-show` will render an element, and use `display:none` to hide it.

Avoid ng-repeat where possible

- It is better to avoid ng-repeat where we can and where it makes sense.
- The `ng-repeat` directive is most likely the worst offender for performance concerns, which means it can easily be abused.
- An ng-repeat likely deals with Arrays of `$scope` Objects and this hammers the `$digest` cycle's performance.

Use batarang to debug

- AngularJS Batarang is a powerful Chrome extension that makes your developer tools “Angular aware”.

- It helps in debugging and profiling AngularJS applications.

Limit HTTP requests

- Every response to a http request will cause a digest cycle to slow down
- A way to limit them would be to create endpoints to do bulk requests when needed, changing multiple indexes in one call, requiring only one digest cycle.

Use $watchCollection instead of $watch

- $watch performs deep checking of an object.
- For deeply nested objects, this can be expensive.
- A better alternative is to use $watchCollection, which limits deep checking to the first layer (which in most cases is enough) and could make major improvements to your performance.

Avoid using filters if at all possible.

- Filters in Angular massively contribute to slow performance.
- Removing filters in the DOM actually impacts $digest cycles, which improves the performance.

Thank You