Dynamic queries flag migration

What does this migration do?

In Angular version 8, a schematic added `static` flags to all `@ViewChild()` and `@ContentChild()` queries. This was the first step towards changing the default behavior. With version 9, the default value changes to `static: false` and the flag becomes optional.

This schematic scans classes in the compilation and for each class, checks if the members have a `@ViewChild()` or `@ContentChild()` query with the `static` flag set to `false`. If so, the schematic removes the flag, as it now matches the default.

Before:

```
@ViewChild('foo', {static: false}) foo: ElementRef;

@ViewChild('bar', {static: true}) bar: ElementRef;
```

After:

```
@ViewChild('foo') foo: ElementRef;

// this query doesn't change because the static value is true
@ViewChild('bar', {static: true}) bar: ElementRef;
```

Note that the flag is not supported in `@ViewChildren()` or `@ContentChildren()` queries, so the schematic will not check these properties.

Why is this migration necessary?

This schematic performs a code cleanup to remove `static` flags that match the default, as they are no longer necessary. Functionally, the code change should be a noop.

Before version 9, Angular figured out the static or dynamic nature of a query automatically, based on how the template was written. Looking at templates in this way, however, caused buggy and surprising behavior (see more about that in the Static Query Migration Guide). As of version 9, Angular uses dynamic queries (`static: false`) by default, which simplifies queries. Developers can still explicitly set a query to `static: true` if necessary.

What is the difference between static and dynamic queries?

The `static` option for `@ViewChild()` and `@ContentChild()` queries determines when the query results become available.

With static queries (`static: true`), the query resolves once the view has been created, but before change detection runs. The result, though, will never be updated to reflect changes to your view, such as changes to `ngIf` and `ngFor` blocks.

With dynamic queries (`static: false`), the query resolves after either `ngAfterViewInit()` or `ngAfterContentInit()` for `@ViewChild()` and `@ContentChild()` respectively. The result will be updated for changes to your view, such as changes to `ngIf` and `ngFor` blocks.

For more information, see the following entries in the Static Query Migration Guide:

- How do I choose which `static` flag value to use: `true` or `false`?
- Is there a case where I should use `{static: true}`?

What does this mean for libraries?

In order to support applications that are still running with version 8, the safest option for libraries is to retain the `static` flag to keep the resolution timing consistent.

- *Libraries on version 9 with applications running version 8: *

  The schematic won't run on libraries. As long as libraries retain their `static` flags from version 8, they should work with apps on 8.

- *Libraries on version 8 with applications running version 9: *

  Libraries will have explicit flags defined. The behavior with explicit flags has not changed.

What about applications using non-migrated libraries?

Because this is a code cleanup that is a noop, non-migrated libraries will work the same either way.