View caching in ionic

(The bellow has been copied/adjusted from the ionic docs: http://ionicframework.com/docs/api/directive/ionNavView/)

- By default, views are cached to improve performance.
- When a view is navigated away from a view, its element is left in the DOM, and its scope (controller) is disconnected from the $watch cycle.
- When navigating to a view that is already cached, its scope (controller) is then reconnected, and the existing element that was left in the DOM becomes the active view.
- This allows for the scroll position of previous views to be maintained. So if you got from a big list into a detail page and then back again, with caching enabled the user will end up in the same place in the big list, w/out caching enabled the user might end up at the top of the big list.

Controllers may only be loaded once!

- When we activate a cached view we are not going to create a controller, the old controller is just re-attached to the element and added back into the angular watch cycle.
- This means that **code in the body of a controller will not be executed again**when entering a cached view.
- This is often a cause for confusion since people expect code in the body of controller to be executed each time

```
app.controller('Controller', function($scope) {
  // Code placed here will only be run once, even if the view is loaded again!
});
```

- Sometimes you need to call some code every-time a view is shown regardless of if it's been loaded up from a cache or created fresh.
- To achieve this you can listen to view lifecycle events , to see the full list of events go to http://ionicframework.com/docs/api/directive/ionView/ and see the section entitled **View LifeCycle and Events**
- Typically we would listen to the $ionicView.enter in our controller.

```
app.controller('Controller', function($scope) {

  $scope.$on('$ionicView.enter', function() {
    // Perform initialisation logic here
  });

});
```

Enabling or Disabling Caching

- Caching can be disabled and enabled in multiple ways.
- By default, Ionic will cache a maximum of 10 views, and not only can this be configured, but apps can also explicitly state which views should and should not be cached.

Disable cache globally

- The $ionicConfigProvider can be used to set the maximum allowable views which can be cached, but this can also be use to disable all caching by setting it to 0.

```
app.config(function($ionicConfigProvider) {
   $ionicConfigProvider.views.maxCache(0);
});
```

Disable cache within state provider

```
$stateProvider.state('myState', {
   cache: false,
   url : '/myUrl',
   templateUrl : 'my-template.html'
})
```

Disable cache with an attribute

```
<ion-view cache-view="false" view-title="My Title!">
  ...
</ion-view>
```

Forward vs Backward Caching

- By **default** caching only works when moving backwards up the history stack, e.g. when moving from

`ListView -> DetailView`

Then DetailView is active and ListView is cached.

If you then go back to `ListView` it's re-attached from the cache however 'DetailView' is not cached.

To switch on **ForwardCaching** you have to define in a config block.

```
app.config(function($ionicConfigProvider) {
   $ionicConfigProvider.views.forwardCache(true);
});
```