$templateCache

The first time a template is used, it is loaded in the template cache for quick retrieval. You can load templates directly into the cache in a script tag, or by consuming the $templateCache service directly.

Adding via the script tag:

<script type="text/ng-template" id="templateId.html">

<p>This is the content of the template</p>

</script>

**Note:** the script tag containing the template does not need to be included in the head of the document, but it must be a descendent of the [$rootElement](https://docs.angularjs.org/api/ng/service/$rootElement) (IE, element with ng-app attribute), otherwise the template will be ignored.

Adding via the $templateCache service:

var myApp = angular.module('myApp', []);

myApp.run(function($templateCache) {

$templateCache.put('templateId.html', 'This is the content of the template');

});

To retrieve the template later, simply use it in your HTML:

<div ng-include=" 'templateId.html' "></div>

or get it via Javascript:

$templateCache.get('templateId.html')

While creating our custom directives, at a certain point, our template code inside the directive may grow complex. To keep it all organized, templates can be linked to the directives differently. For this, we have some alternatives:  
  
Creating another file, of course and linking it to our directives.  
  
For this, all we need to do is yank our template code and paste it inside another HTML, file say "template.html".  
  
Linking this file to our directive can be done this way:

1. app.directive('myDir', **function**()
2. {
3. **return**
4. {
5. templateUrl: '~path-to-file/template.html'};
6. });

Another way to organize our template code outside the directive is to place the template code in our index.html only, within "script" tags. We'll see how.

1. <div ng-app='myApp'>
2. <script type=”text/ng-template” id=”template.html”> <! – template code here --> </script>
3. <input my-dir type=”text” .. />
4. </div>

Both of these ways will work exactly the same way it used to work when we wrote all the template code inside the directive only.  
  
**$templateCache**  
  
All the template code that we link to our directive resides inside the $templateCache. This $templateCache can be manipulated to "put" and "get" template codes from it.  
  
Long story short, if we remove the template code in the script tag and delete the template file too, the template's code can still be maintained using the $templateCache. See how:

1. app.run( **function** ($templateCache)
2. {
3. $templateCache.put(“template.html”, “<div>This is template **for** the directive</div>”);
4. });

We just "put" the template code inside the $templateCache identified by a key "template.html" and we will use it further to get the code by this key inside our directive. We do this by injecting $templateCache into it:

1. app.directive('myDir', **function**($templateCache)
2. {
3. **return**
4. {
5. template: $templateCache.get('template.html') };
6. });

This is still going to work the same as it kept working for us from the start.