<https://github.com/angular-ui/ui-router/wiki/Quick-Reference#wiki-stategoto--toparams--options>

注意事项：

$state.go(to,[,toParams][,options]) 内部调用了$state.transitionTo()，但是$state.go更加全面，推荐使用$state.go().

1. $state.go() 跳转
2. $state.transitionTo() 跳转 （推荐使用$state.go()）
3. $state.reload() 刷新当前页

相当于

$state.transitionTo($state.current, $stateParams, {

reload: true, inherit: false, notify: false

});

1. $state.includes(stateName [,params]) 是否被嵌套包含在stateName里面(包括自己)

For example, if you had the following states set up:

* contacts
* contacts.list
* contacts.details
* contacts.details.item
* about

So, e.g. if you were within contacts.details.item then:

$state.includes("contacts"); // returns true

$state.includes("contacts.details"); // returns true

$state.includes("contacts.details.item"); // returns true

$state.includes("contacts.list"); // returns false

$state.includes("about"); // returns false

1. $state.is(stateOrName [,params]) //判断是否当前$state是否就是stateOrName。
2. $state.href(stateOrName [,params] [,options])

expect($state.href("about.person", { person: "bob" })).toEqual("/about/bob");

1. $state.get([stateName])

$state.get(stateName) //获取stateName的配置object

$state.get() //获取数组of 所有的state 配置object

1. $state.current 当前的state config object

# **$state**

## **$state.go(to [, toParams] [, options])**

Returns a ****Promise**** representing the state of the transition.

Convenience method for transitioning to a new state. $state.go calls [$state.transitionTo](https://github.com/angular-ui/ui-router/wiki/" \l "wiki-statetransitiontoto--toparams--updatelocationoptions)internally but automatically sets options to { location: true, inherit: true, relative: $state.$current, notify: true }. This allows you to easily use an absolute or relative to path and specify only the parameters you'd like to update (while letting unspecified parameters inherit from the current state.

### **to**

****String**** Absolute State Name or Relative State Path

The name of the state that will be transitioned to ****or**** a relative state path. If the path starts with ^or . then it is relative, otherwise it is absolute.

Some examples:

* $state.go('contact.detail') will go to the 'contact.detail' state
* $state.go('^') will go to a parent state.
* $state.go('^.sibling') will go to a sibling state.
* $state.go('.child') will go to a child state.
* $state.go('.child.grandchild') will go to a grandchild state.

### **toParams**

****Object****

A map of the parameters that will be sent to the state, will populate [$stateParams](https://github.com/angular-ui/ui-router/wiki/" \l "wiki-stateparams).

Any parameters that are not specified will be inherited from currently defined parameters. This allows, for example, going to a sibling state that shares parameters specified in a parent state. Parameter inheritance only works between common ancestor states, I.e. transitioning to a sibling will get you the parameters for all parents, transitioning to a child will get you all current parameters, etc.

### **options**

****Object****

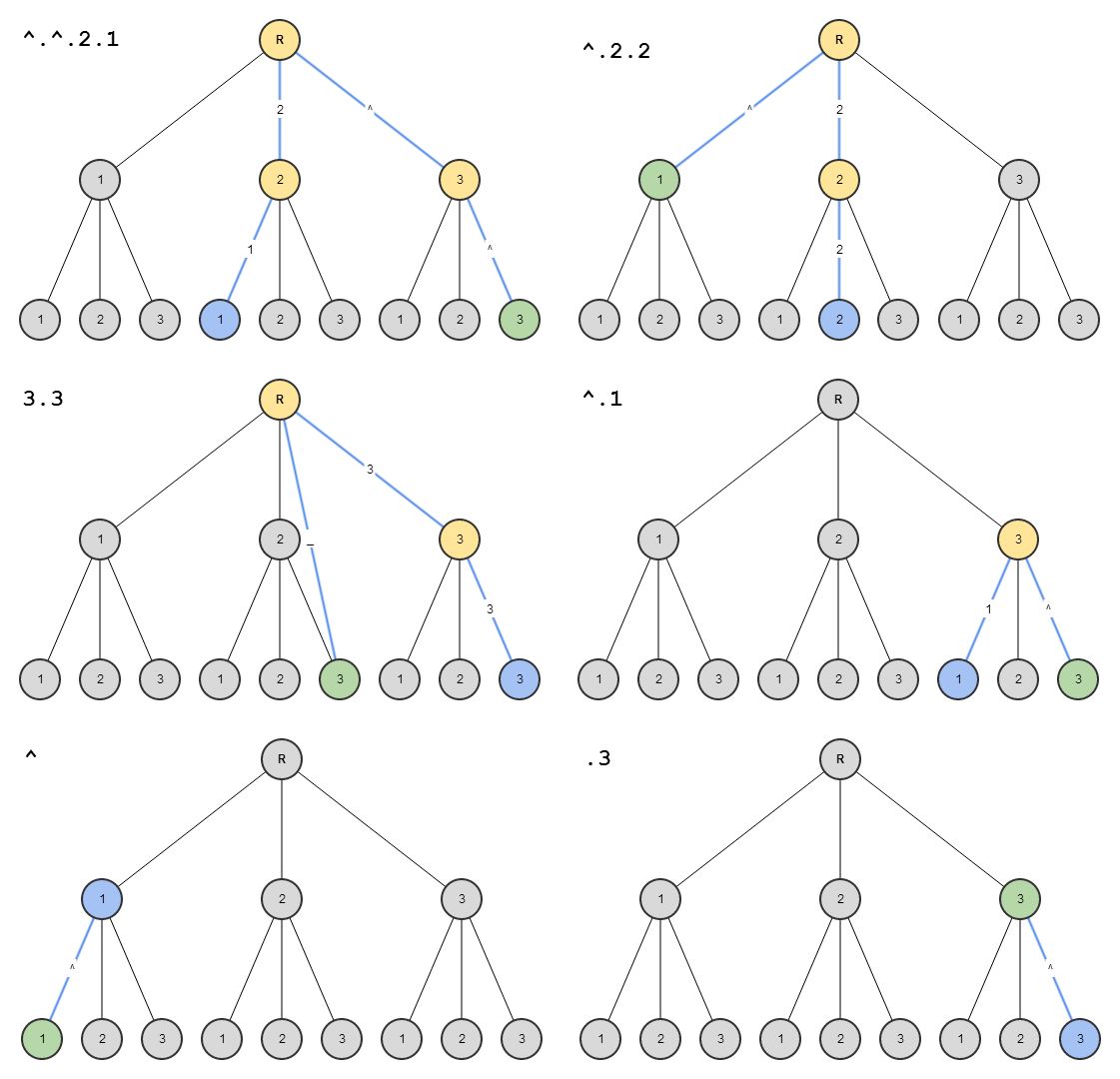
If Object is passed, object is an options hash. The following options are supported:

* location Boolean or "replace" (default true), If true will update the url in the location bar, iffalse will not. If string "replace", will update url and also replace last history record.
* inherit Boolean (default true), If true will inherit url parameters from current url.
* relative stateObject (default $state.$current), When transitioning with relative path (e.g '^'), defines which state to be relative from.
* notify Boolean (default true), If true will broadcast $stateChangeStart and $stateChangeSuccess events.
* reload v0.2.5 Boolean (default false), If true will force transition even if the state or params have not changed, aka a reload of the same state. It differs from reloadOnSearch because you'd use this when you want to force a reload when everything is the same, including search params.

### **Examples Diagram:**

* Green = Starting State
* Yellow = Intermediary State
* Blue = Final Destination State

[Enlarge](https://raw.github.com/wiki/angular-ui/ui-router/StateGoExamples.png)



## **$state.transitionTo(to, toParams [, options])**

Returns a ****Promise**** representing the state of the transition.

Low-level method for transitioning to a new state. [$state.go()](https://github.com/angular-ui/ui-router/wiki/" \l "wiki-stategoto--toparams--options) uses transitionTo internally.

**[$state.go()](https://github.com/angular-ui/ui-router/wiki/" \l "wiki-stategoto--toparams--options) is recommended in most situations.**

### **to**

****String****

The name of the state that will be transitioned to.

### **toParams**

****Object****

A map of the parameters that will be sent to the state, will populate [$stateParams](https://github.com/angular-ui/ui-router/wiki/" \l "wiki-stateparams).

### **options**

****Object****

Object is an options hash. The following options are supported:

* location Boolean or "replace" (default true), If true will update the url in the location bar, iffalse will not. If string "replace", will update url and also replace last history record.
* inherit Boolean (default false), If true will inherit url parameters from current url.
* relative stateObject (default null), When transitioning with relative path (e.g '^'), defines which state to be relative from.
* notify Boolean (default true), If true will broadcast $stateChangeStart and $stateChangeSuccess events.
* reload v0.2.5 Boolean (default false), If true will force transition even if the state or params have not changed, aka a reload of the same state. It differs from reloadOnSearch because you'd use this when you want to force a reload when everything is the same, including search params.

## **$state.reload()**

Returns ****null****

A method that force reloads the current state. All resolves are re-resolved, events are not re-fired, and controllers reinstantiated (bug with controllers reinstantiating right now, fixing soon).

This is just an alias for:

$state.transitionTo($state.current, $stateParams, {

reload: true, inherit: false, notify: false

});

## **$state.includes(stateName [, params])**

Returns ****Boolean****

A method to determine if the current active state is equal to or is the child of the state stateName. If any params are passed then they will be tested for a match as well. Not all the parameters need to be passed, just the ones you'd like to test for equality.

### **stateName**

****String****

A partial name to be searched for within the current state name. For example, if you had the following states set up:

* contacts
* contacts.list
* contacts.details
* contacts.details.item
* about

So, e.g. if you were within contacts.details.item then:

$state.includes("contacts"); // returns true$state.includes("contacts.details"); // returns true$state.includes("contacts.details.item"); // returns true$state.includes("contacts.list"); // returns false$state.includes("about"); // returns false

### **params (v0.3)**

****Object****

A param object, e.g. {sectionId: section.id}, that you'd like to test against the current active state.

Let's say the current active state was "contacts.details.item.edit" activated by a url of "/contacts/1/address/edit", where 1 populates the :id param and 'address' populates the:item param. So then:

$state.includes("contacts.detail", {id: 1}); // returns true$state.includes("contacts.detail.item", {item:'address'}); // returns true$state.includes("contacts", {bogus:'gnarly'}); // returns false

## **$state.is(stateOrName [, params])**

Returns ****Boolean****

Similar to includes, but only checks for the full state name. If params is supplied then it will be tested for strict equality against the current active params object, so all params must match with none missing and no extras.

### **stateOrName**

****String**** or ****Object****

The state name or state object you'd like to check.

So, e.g. if you were within contact.details.item then:

$state.is("contact.details.item"); // returns true$state.is(contactDetailItemStateConfigObj); // returns true// Everything else would return false

### **params (v0.3)**

****Object****

A param object, e.g. {sectionId: section.id}, that you'd like to test against the current active state.

Let's say the current active state was "contacts.details.item.edit" activated by a url of "/contacts/1/address/edit", where 1 populates the :id param and 'address' populates the:item param. So then:

$state.includes("contacts.detail.item.edit", {id: 1, item: 'address'}); // returns true// Everything else returns `false`

## **$state.href(stateOrName [, params] [, options])**

Returns ****String**** Compiled URL

A url generation method that returns the compiled url for the given state populated with the given params.

e.g. expect($state.href("about.person", { person: "bob" })).toEqual("/about/bob");

Note: returns null if no valid url can be constructed.

### **stateOrName**

****String**** or ****Object****

The state name or state object you'd like to generate a url from.

### **params**

****Object****

An object of parameter values to fill the state's required parameters.

### **options**

****Object****

An options hash, the following options are available:

* lossy Boolean (default true) If true, and if there is no url associated with the state provided in the first parameter, then the constructed href url will be built from the first navigable ancestor (aka ancestor with a valid url).
* inherit Boolean (default false) If true will inherit url parameters from current url.
* relative stateObject (default $state.$current), When transitioning with relative path (e.g '^'), defines which state to be relative from.
* absolute Boolean (default false) If true will generate an absolute url, e.g."http://www.example.com/fullurl".

## **$state.get([stateName])**

Returns ****Object****

#### **Usages**

* get(stateName) - A method for retrieving the configuration object for any state, by passing the name as a string.
* get() - v0.3.0 - Returns an array of all state config objects.

### **stateName**

****String****

The name of the state for which you'd like to get the original state configuration object for.

## **$state.current**

Returns ****State Object****

A reference to the state's config object. However you passed it in. Useful for accessing custom data.

[Learn More about the state config object](https://github.com/angular-ui/ui-router/wiki/" \l "wiki-stateconfig)