The Promises goal is to allow users to monitor asynchronous progress by providing a "promise" as a return from a call. A number of Angular services return promises: $http, $interval, $timeout

AngularJS event system provides a lot of power to our Angular apps.

Angular use $q library to implement promises.

How promises run

1. Execute $q.defer();
2. Execute resolve()
3. Execute Promises

//Defining module

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

//Define Controller

app.controller("FirstController", function ($scope, $q) {

var defer = $q.defer();

defer.promise

.then(function () {

alert("I am first Method");

})

.then(function () {

alert("I am second Method");

})

.then(function () {

alert("I am third Method");

});

defer.resolve();

$scope.Name = "Angular Promises Demo";

});

How to use output of one method as input to another one

//Defining module

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

//Define Controller

app.controller("FirstController", function ($scope, $q) {

var defer = $q.defer();

defer.promise

.then(function (Name) {

alert("I am first Method" + Name);

return "Method2"

})

.then(function (Name) {

alert("I am second Method" + Name);

return "Method3"

})

.then(function (Name) {

alert("I am third Method" + Name);

});

defer.resolve("Method1");

$scope.Name = "Angular Promises Demo";

});

**Wrap-up**

In short, promises fuel AngularJS asynchronous operations. Anything needing to run in the background will need to coordinate with a caller such as a controller or directive, and the promise API is the way to go.