<https://github.com/oauthjs/angular-oauth2/blob/master/dist/angular-oauth2.js>

Service oauthFixInterceptor.js

angular.module('app.services')  
 .service('oauthFixInterceptor', ['$q', '$rootScope', 'OAuthToken',  
 **function** ($q, $rootScope, OAuthToken) {  
 **return** {  
 request: **function** (config) {  
 config.headers = config.headers || {};  
 **if** (OAuthToken.getAuthorizationHeader()) {  
 config.headers = config.headers || {};  
 config.headers.Authorization = OAuthToken.getAuthorizationHeader();  
 }  
 **return** config;  
 },  
 responseError: **function** (rejection) {  
 **var** deferred = $q.defer();  
 **if** (400 === rejection.status && rejection.data &&  
 ("invalid\_request" === rejection.data.error ||  
 "invalid\_grant" === rejection.data.error)) {  
 OAuthToken.removeToken();  
 $rootScope.$emit("oauth:error", {rejection: rejection, deferred: deferred});  
 }  
 **if** (401 === rejection.status && rejection.data &&  
 "access\_denied" === rejection.data.error ||  
 rejection.headers("www-authenticate") && 0 ===  
 rejection.headers("www-authenticate").indexOf("Bearer")) {  
 $rootScope.$emit("oauth:error", {rejection: rejection, deferred: deferred});  
 **return** deferred.promise;  
 }  
 **return** $q.reject(rejection);  
  
 }  
 };  
 }]);

<script src="{{ asset('build/js/services/oauthFixInterceptor.js') }}"></script>

**function**($routeProvider, $httpProvider, OAuthProvider, OAuthTokenProvider, appConfigProvider){  
 $httpProvider.defaults.headers.post['Content-Type'] = 'application/x-www-form-urlencoded;charset=utf-8';  
 $httpProvider.defaults.headers.put['Content-Type'] = 'application/x-www-form-urlencoded;charset=utf-8';  
 $httpProvider.defaults.transformRequest = appConfigProvider.config.utils.transformRequest;  
 $httpProvider.defaults.transformResponse = appConfigProvider.config.utils.transformResponse;  
 $httpProvider.interceptors.splice(0, 1);  
 $httpProvider.interceptors.splice(0, 1);  
 $httpProvider.interceptors.push('oauthFixInterceptor');

//depois que o angular é carregado isso é executado  
//esta adicionando um evento OAuth error para se for invalido não retornanr se for um token  
// invalido retoranar uma atualizacao o token  
app.run(['$rootScope', '$location', 'OAuth', **function**($rootScope, $location, OAuth) {  
 //evento e proxima rota atual  
 $rootScope.$on('$routeChangeStart', **function**(event,next,current){  
 //olhar next se é diferente do login e se token existe na aplicação  
 //url da rota que o usuario vai acessar  
 **if**(next.$$route.originalPath != '/login') {  
 //vai nos cookies do angular e verifica o token se não existir ele retorna falso  
 **if**(!OAuth.isAuthenticated()) {  
 $location.path('login');  
 }  
 }  
 });  
  
 $rootScope.$on('oauth:error', **function** (event, data) {  
 // Ignore `invalid\_grant` error - should be catched on `LoginController`.  
 **if** ('invalid\_grant' === data.rejection.data.error) {  
 **return**;  
 }  
  
 // Refresh token when a `invalid\_token` error occurs.  
 **if** ('access\_denied' === data.rejection.data.error) {  
 **return** OAuth.getRefreshToken();  
 }  
  
 // Redirect to `/login` with the `error\_reason`.  
 **return** $location.path('/login');  
 });  
}]);