

*Universidad Tecnológica de la*

*Riviera Maya*

**Desarrollo de Aplicaciones III**

**Profesor:**

*Lic. Julián Villegas*

**Autor(es):**

*Domínguez Juárez Juan Antonio*

*Solidaridad, Quintana Roo, 20 de abril del 2018*

**Carrera:**

*TSU en Tecnologías de la Información y Comunicación Área Sistemas Informáticos (TI51)*

**Tabla de contenido**

[**Definición** 2](#_Toc511999860)

[**Ejemplo** 2](#_Toc511999861)

[**Fuentes de Información** 4](#_Toc511999862)

## **Definición**

El objeto $http Es un servicio que viene incorporado con Angular, y que permite comunicarnos con servidores HTTP remotos, a través de pedidos asincrónicos que transmiten datos

* utilizando el objeto [XMLHttpRequest](https://developer.mozilla.org/en/xmlhttprequest) del browser
* o vía [JSONP](http://en.wikipedia.org/wiki/JSONP)

Se inyecta en el constructor de cada service que necesite hacer una llamada remota.

## **Ejemplo**

Vemos a continuación un service de Angular que hace dos llamadas al server en la aplicación de Tareas pendientes

* en un método http GET lista las tareas (pendientes o todas, eso lo define el servicio REST en el server)
* en un método http PUT actualiza la tarea (por ejemplo, para asignarla o indicar que está cumplida)

app.service('tareasService', function ($http) {

this.findAll **=** function (callback, errorHandler) {

**$**http.get('/tareas').then(callback, errorHandler);

};

this.update **=** function (tarea, callback, errorHandler) {

**$**http.put('/tareas/' **+** tarea.id, tarea).then(callback, errorHandler);

};

});

El lector puede apreciar cómo el objeto $http se inyecta como referencia en la función que define el comportamiento del service.

Cada llamada http necesita pasar dos callbacks mediante el mensaje then:

* el primer parámetro es una función que indica lo que debe hacer cuando la operación se complete
* el segundo parámetro es otra función que indica cómo debe comportarse el cliente en caso de error en el servidor (código distinto de 200 = OK)

En el callback se recibe la respuesta del server (objeto response), con el header y el contenido o data. Para llenar la lista de tareas el controller debe hacer lo siguiente:

app.controller('TareasController', function (tareasService, $timeout) {

...

// TRAER LAS TAREAS

this.getTareas **=** function () {

tareasService.findAll(function (response) {

self.tareas **=** response.data;

}, notificarError);

}

## **Fuentes de Información**

<http://algo3.uqbar-project.org/material/herramientas/angular/angularjs---http-server>

<http://blog.enriqueoriol.com/2017/11/httpclient-vs-http-angular.html>