// directive

app.directive('fileUpload', function () {  
 return {  
 scope: true, //create a new scope  
 link: function (scope, el, attrs) {  
 el.bind('change', function (event) {  
 var files = event.target.files;  
 //iterate files since 'multiple' may be specified on the element  
 for (var i = 0;i<files.length;i++) {  
 //emit event upward  
 scope.$emit("fileSelected", { file: files[i] });  
 }   
 });  
 }  
 };  
});

// html

<div ng-controller="Ctrl">

<input type="file" file-upload multiple/>

<ul>  
 <li ng-repeat="file in files">{{file.name}}</li>  
 </ul>  
</div>

// controller

function Ctrl($scope, $http) {  
  
 //a simple model to bind to and send to the server  
 $scope.model = {  
 name: "",  
 comments: ""  
 };  
  
 //an array of files selected  
 $scope.files = [];  
  
 //listen for the file selected event  
 $scope.$on("fileSelected", function (event, args) {  
 $scope.$apply(function () {   
 //add the file object to the scope's files collection  
 $scope.files.push(args.file);  
 });  
 });  
   
 //the save method  
 $scope.save = function() {  
 $http({  
 method: 'POST',  
 url: "/Api/PostStuff",  
 //IMPORTANT!!! You might think this should be set to 'multipart/form-data'   
 // but this is not true because when we are sending up files the request   
 // needs to include a 'boundary' parameter which identifies the boundary   
 // name between parts in this multi-part request and setting the Content-type   
 // manually will not set this boundary parameter. For whatever reason,   
 // setting the Content-type to 'false' will force the request to automatically  
 // populate the headers properly including the boundary parameter.  
 headers: { 'Content-Type': false },  
 //This method will allow us to change how the data is sent up to the server  
 // for which we'll need to encapsulate the model data in 'FormData'  
 transformRequest: function (data) {  
 var formData = new FormData();  
 //need to convert our json object to a string version of json otherwise  
 // the browser will do a 'toString()' on the object which will result   
 // in the value '[Object object]' on the server.  
 formData.append("model", angular.toJson(data.model));  
 //now add all of the assigned files  
 for (var i = 0; i < data.files; i++) {  
 //add each file to the form data and iteratively name them  
 formData.append("file" + i, data.files[i]);  
 }  
 return formData;  
 },  
 //Create an object that contains the model and files which will be transformed  
 // in the above transformRequest method  
 data: { model: $scope.model, files: $scope.files }  
 }).  
 success(function (data, status, headers, config) {  
 alert("success!");  
 }).  
 error(function (data, status, headers, config) {  
 alert("failed!");  
 });  
 };  
};

// web api

public async Task<HttpResponseMessage> PostStuff()  
{  
 if (!Request.Content.IsMimeMultipartContent())  
 {  
 throw new HttpResponseException(HttpStatusCode.UnsupportedMediaType);  
 }  
  
 var root = HttpContext.Current.Server.MapPath("~/App\_Data/Temp/FileUploads");  
 Directory.CreateDirectory(root);  
 var provider = new MultipartFormDataStreamProvider(root);  
 var result = await Request.Content.ReadAsMultipartAsync(provider);  
 if (result.FormData["model"] == null)  
 {  
 throw new HttpResponseException(HttpStatusCode.BadRequest);  
 }  
  
 var model = result.FormData["model"];  
 //TODO: Do something with the json model which is currently a string

//get the files  
 foreach (var file in result.FileData)  
 {   
 //TODO: Do something with each uploaded file  
 }  
  
 return Request.CreateResponse(HttpStatusCode.OK, "success!");  
}

<ul>  
 <li ng-repeat="file in files">{{file.name}}</li>  
 </ul>  
</div>