**<!DOCTYPE HTML>**

**<html>**

**<head>**

**<meta http-equiv="Content-Type" content="text/html; charset=utf-8">**

**<meta name="viewport" content="initial-scale=1, maximum-scale=1,user-scalable=no">**

**<title>File Access with Drag and Drop</title>**

**<link rel="stylesheet" href="https://js.arcgis.com/3.15/dijit/themes/claro/claro.css">**

**<link rel="stylesheet" href="https://js.arcgis.com/3.15/esri/css/esri.css" />**

**<link rel="stylesheet" href="css/layout.css" />**

**<script src="https://js.arcgis.com/3.15/"></script>**

**<script>**

**var map;**

**require([**

**"esri/config",**

**"esri/domUtils",**

**"esri/graphic",**

**"esri/InfoTemplate",**

**"esri/map",**

**"esri/request",**

**"esri/urlUtils",**

**"esri/dijit/InfoWindowLite",**

**"esri/geometry/Multipoint",**

**"esri/geometry/Point",**

**"esri/geometry/webMercatorUtils",**

**"esri/layers/ArcGISDynamicMapServiceLayer",**

**"esri/layers/ArcGISImageServiceLayer",**

**"esri/layers/FeatureLayer",**

**"esri/symbols/PictureMarkerSymbol",**

**"dojo/dom",**

**"dojo/dom-construct",**

**"dojo/json",**

**"dojo/on",**

**"dojo/parser",**

**"dojo/\_base/array",**

**"dojo/\_base/lang",**

**"dojox/data/CsvStore",**

**"dojox/encoding/base64",**

**"dijit/Dialog",**

**"dijit/layout/BorderContainer",**

**"dijit/layout/ContentPane",**

**"dojo/domReady!"**

**],**

**function (**

**esriConfig, domUtils, Graphic, InfoTemplate, Map, request, urlUtils,**

**InfoWindowLite, Multipoint, Point, webMercatorUtils, ArcGISDynamicMapServiceLayer,**

**ArcGISImageServiceLayer, FeatureLayer, PictureMarkerSymbol, dom, domConstruct,**

**JSON, on, parser, arrayUtils, lang, CsvStore, base64**

**) {**

**parser.parse();**

**//list of lat and lon field strings**

**var latFieldStrings = ["lat", "latitude", "y", "ycenter"];**

**var longFieldStrings = ["lon", "long", "longitude", "x", "xcenter"];**

**//specfiy cors enabled server and proxy for backup**

**esriConfig.defaults.io.corsEnabledServers.push("serverapi.arcgisonline.com");**

**esriConfig.defaults.io.proxyUrl = 'http://serverapi.arcgisonline.com/proxy/proxy.ashx';**

**on(dom.byId("clearButton"), "click", clearAll);**

**on(dom.byId("uploadForm"), "change", function () {**

**uploadFile(this.files);**

**});**

**setupDropZone();**

**map = new Map("mapCanvas", {**

**basemap: "topo",**

**center: [-103.272, 39.096],**

**zoom: 4,**

**slider: false**

**});**

**map.infoWindow.resize(275, 175);**

**function setupDropZone () {**

**// Let's verify that we have proper browser support, before**

**// moving ahead. You can also use a library like Modernizr**

**// to detect browser capabilities:**

**// http://www.modernizr.com/**

**if (!window.File || !window.FileReader) {**

**domUtils.show(dom.byId('uploadForm'));**

**domUtils.show(dom.byId('msg'));**

**return;**

**}**

**var mapCanvas = dom.byId("mapCanvas");**

**// Reference**

**// http://www.html5rocks.com/features/file**

**// http://www.html5rocks.com/tutorials/dnd/basics/**

**// https://developer.mozilla.org/En/DragDrop/Drag\_Operations**

**on(mapCanvas, "dragenter", function (event) {**

**// If we don't prevent default behavior here, browsers will**

**// perform the default action for the file being dropped i.e,**

**// point the page to the file.**

**event.preventDefault();**

**});**

**on(mapCanvas, "dragover", function (event) {**

**event.preventDefault();**

**});**

**on(mapCanvas, "drop", handleDrop);**

**}**

**function handleDrop (event) {**

**console.log("Drop: ", event);**

**event.preventDefault();**

**// Reference**

**// http://www.html5rocks.com/tutorials/file/dndfiles/**

**// https://developer.mozilla.org/en/Using\_files\_from\_web\_applications**

**var dataTransfer = event.dataTransfer,**

**files = dataTransfer.files,**

**types = dataTransfer.types;**

**// File drop?**

**if (files && files.length === 1) {**

**console.log("[ FILES ]");**

**var file = files[0]; // that's right I'm only reading one file**

**console.log("type = ", file.type);**

**if (file.type.indexOf("image/") !== -1) {**

**handleImage(file, event.layerX, event.layerY);**

**}**

**else if (file.name.indexOf(".csv") !== -1) {**

**handleCSV(file);**

**}**

**}**

**// Textual drop?**

**else if (types) {**

**console.log("[ TYPES ]");**

**console.log(" Length = ", types.length);**

**arrayUtils.forEach(types, function (type) {**

**if (type) {**

**console.log(" Type: ", type);**

**console.log(" Data: ", dataTransfer.getData(type));**

**}**

**});**

**// We're looking for URLs only.**

**var url;**

**arrayUtils.some(types, function (type) {**

**if (type.indexOf("text/uri-list") !== -1) {**

**url = dataTransfer.getData("text/uri-list");**

**return true;**

**}**

**else if (type.indexOf("text/x-moz-url") !== -1) {**

**url = dataTransfer.getData("text/plain");**

**return true;**

**}**

**else if (type.indexOf("text/plain") !== -1) {**

**url = dataTransfer.getData("text/plain");**

**url = url.replace(/^\s+|\s+$/g, "");**

**if (url.indexOf("http") === 0) {**

**return true;**

**}**

**}**

**return false;**

**});**

**if (url) {**

**url = url.replace(/^\s+|\s+$/g, "");**

**// Check if this URL is a google search result.**

**// If so, parse it and extract the actual URL**

**// to the search result**

**if (url.indexOf("www.google.com/url") !== -1) {**

**var obj = urlUtils.urlToObject(url);**

**if (obj && obj.query && obj.query.url) {**

**url = obj.query.url;**

**}**

**}**

**if (url.match(/MapServer\/?$/i)) {**

**// ArcGIS Server Map Service?**

**handleMapServer(url);**

**}**

**else if (url.match(/(Map|Feature)Server\/\d+\/?$/i)) {**

**// ArcGIS Server Map/Feature Service Layer?**

**handleFeatureLayer(url);**

**}**

**else if (url.match(/ImageServer\/?$/i)) {**

**// ArcGIS Server Image Service?**

**handleImageService(url);**

**}**

**}**

**}**

**}**

**function handleImage (file, x, y) {**

**console.log("Processing IMAGE: ", file, ", ", file.name, ", ", file.type, ", ", file.size);**

**var reader = new FileReader();**

**reader.onload = function () {**

**console.log("Finished reading the image");**

**// Create an image element just to find out the image**

**// dimension before adding it as a graphic**

**var img = domConstruct.create("img");**

**img.onload = function () {**

**var width = img.width,**

**height = img.height;**

**console.log("Image dimensions: ", width, ", ", height);**

**// Add a graphic with this image as its symbol**

**var symbol = new PictureMarkerSymbol(reader.result,**

**width > 64 ? 64 : width,**

**height > 64 ? 64 : height);**

**var point = map.toMap(new Point(x, y));**

**var graphic = new Graphic(point, symbol);**

**map.graphics.add(graphic);**

**};**

**img.src = reader.result;**

**};**

**// Note that it's possible to monitor read progress as well:**

**// http://www.html5rocks.com/tutorials/file/dndfiles/#toc-monitoring-progress**

**// http://www.html5rocks.com/tutorials/file/dndfiles/#toc-reading-files**

**reader.readAsDataURL(file);**

**}**

**function handleMapServer (url) {**

**console.log("Processing MS: ", url);**

**var layer = new ArcGISDynamicMapServiceLayer(url, {**

**opacity: 0.75**

**});**

**map.addLayer(layer);**

**}**

**function handleFeatureLayer (url) {**

**console.log("Processing FL: ", url);**

**var layer = new FeatureLayer(url, {**

**opacity: 0.75,**

**mode: FeatureLayer.MODE\_ONDEMAND,**

**infoTemplate: new InfoTemplate(null, "${\*}")**

**});**

**map.addLayer(layer);**

**}**

**function handleImageService (url) {**

**console.log("Processing IS: ", url);**

**var layer = new ArcGISImageServiceLayer(url, {**

**opacity: 0.75**

**});**

**map.addLayer(layer);**

**}**

**function handleCSV (file) {**

**console.log("Processing CSV: ", file, ", ", file.name, ", ", file.type, ", ", file.size);**

**if (file.data) {**

**var decoded = bytesToString(base64.decode(file.data));**

**processCSVData(decoded);**

**}**

**else {**

**var reader = new FileReader();**

**reader.onload = function () {**

**console.log("Finished reading CSV data");**

**processCSVData(reader.result);**

**};**

**reader.readAsText(file);**

**}**

**}**

**var bytesToString = function (b) {**

**console.log("bytes to string");**

**var s = [];**

**arrayUtils.forEach(b, function (c) {**

**s.push(String.fromCharCode(c));**

**});**

**return s.join("");**

**};**

**function processCSVData (data) {**

**var newLineIndex = data.indexOf("\n");**

**var firstLine = lang.trim(data.substr(0, newLineIndex)); //remove extra whitespace, not sure if I need to do this since I threw out space delimiters**

**var separator = getSeparator(firstLine);**

**var csvStore = new CsvStore({**

**data: data,**

**separator: separator**

**});**

**csvStore.fetch({**

**onComplete: function (items) {**

**var objectId = 0;**

**var featureCollection = generateFeatureCollectionTemplateCSV(csvStore, items);**

**var popupInfo = generateDefaultPopupInfo(featureCollection);**

**var infoTemplate = new InfoTemplate(buildInfoTemplate(popupInfo));**

**var latField, longField;**

**var fieldNames = csvStore.getAttributes(items[0]);**

**arrayUtils.forEach(fieldNames, function (fieldName) {**

**var matchId;**

**matchId = arrayUtils.indexOf(latFieldStrings,**

**fieldName.toLowerCase());**

**if (matchId !== -1) {**

**latField = fieldName;**

**}**

**matchId = arrayUtils.indexOf(longFieldStrings,**

**fieldName.toLowerCase());**

**if (matchId !== -1) {**

**longField = fieldName;**

**}**

**});**

**// Add records in this CSV store as graphics**

**arrayUtils.forEach(items, function (item) {**

**var attrs = csvStore.getAttributes(item),**

**attributes = {};**

**// Read all the attributes for this record/item**

**arrayUtils.forEach(attrs, function (attr) {**

**var value = Number(csvStore.getValue(item, attr));**

**attributes[attr] = isNaN(value) ? csvStore.getValue(item, attr) : value;**

**});**

**attributes["\_\_OBJECTID"] = objectId;**

**objectId++;**

**var latitude = parseFloat(attributes[latField]);**

**var longitude = parseFloat(attributes[longField]);**

**if (isNaN(latitude) || isNaN(longitude)) {**

**return;**

**}**

**var geometry = webMercatorUtils**

**.geographicToWebMercator(new Point(longitude, latitude));**

**var feature = {**

**"geometry": geometry.toJson(),**

**"attributes": attributes**

**};**

**featureCollection.featureSet.features.push(feature);**

**});**

**var featureLayer = new FeatureLayer(featureCollection, {**

**infoTemplate: infoTemplate,**

**id: 'csvLayer'**

**});**

**featureLayer.\_\_popupInfo = popupInfo;**

**map.addLayer(featureLayer);**

**zoomToData(featureLayer);**

**},**

**onError: function (error) {**

**console.error("Error fetching items from CSV store: ", error);**

**}**

**});**

**}**

**function generateFeatureCollectionTemplateCSV (store, items) {**

**//create a feature collection for the input csv file**

**var featureCollection = {**

**"layerDefinition": null,**

**"featureSet": {**

**"features": [],**

**"geometryType": "esriGeometryPoint"**

**}**

**};**

**featureCollection.layerDefinition = {**

**"geometryType": "esriGeometryPoint",**

**"objectIdField": "\_\_OBJECTID",**

**"type": "Feature Layer",**

**"typeIdField": "",**

**"drawingInfo": {**

**"renderer": {**

**"type": "simple",**

**"symbol": {**

**"type": "esriPMS",**

**"url": "http://static.arcgis.com/images/Symbols/Basic/RedSphere.png",**

**"imageData": "",**

**"contentType": "image/png",**

**"width": 15,**

**"height": 15**

**}**

**}**

**},**

**"fields": [**

**{**

**"name": "\_\_OBJECTID",**

**"alias": "\_\_OBJECTID",**

**"type": "esriFieldTypeOID",**

**"editable": false,**

**"domain": null**

**}**

**],**

**"types": [],**

**"capabilities": "Query"**

**};**

**var fields = store.getAttributes(items[0]);**

**arrayUtils.forEach(fields, function (field) {**

**var value = store.getValue(items[0], field);**

**var parsedValue = Number(value);**

**if (isNaN(parsedValue)) { //check first value and see if it is a number**

**featureCollection.layerDefinition.fields.push({**

**"name": field,**

**"alias": field,**

**"type": "esriFieldTypeString",**

**"editable": true,**

**"domain": null**

**});**

**}**

**else {**

**featureCollection.layerDefinition.fields.push({**

**"name": field,**

**"alias": field,**

**"type": "esriFieldTypeDouble",**

**"editable": true,**

**"domain": null**

**});**

**}**

**});**

**return featureCollection;**

**}**

**function generateDefaultPopupInfo (featureCollection) {**

**var fields = featureCollection.layerDefinition.fields;**

**var decimal = {**

**'esriFieldTypeDouble': 1,**

**'esriFieldTypeSingle': 1**

**};**

**var integer = {**

**'esriFieldTypeInteger': 1,**

**'esriFieldTypeSmallInteger': 1**

**};**

**var dt = {**

**'esriFieldTypeDate': 1**

**};**

**var displayField = null;**

**var fieldInfos = arrayUtils.map(fields,**

**lang.hitch(this, function (item) {**

**if (item.name.toUpperCase() === "NAME") {**

**displayField = item.name;**

**}**

**var visible = (item.type !== "esriFieldTypeOID" &&**

**item.type !== "esriFieldTypeGlobalID" &&**

**item.type !== "esriFieldTypeGeometry");**

**var format = null;**

**if (visible) {**

**var f = item.name.toLowerCase();**

**var hideFieldsStr = ",stretched value,fnode\_,tnode\_,lpoly\_,rpoly\_,poly\_,subclass,subclass\_,rings\_ok,rings\_nok,";**

**if (hideFieldsStr.indexOf("," + f + ",") > -1 ||**

**f.indexOf("area") > -1 || f.indexOf("length") > -1 ||**

**f.indexOf("shape") > -1 || f.indexOf("perimeter") > -1 ||**

**f.indexOf("objectid") > -1 || f.indexOf("\_") == f.length - 1 ||**

**f.indexOf("\_i") == f.length - 2) {**

**visible = false;**

**}**

**if (item.type in integer) {**

**format = {**

**places: 0,**

**digitSeparator: true**

**};**

**}**

**else if (item.type in decimal) {**

**format = {**

**places: 2,**

**digitSeparator: true**

**};**

**}**

**else if (item.type in dt) {**

**format = {**

**dateFormat: 'shortDateShortTime'**

**};**

**}**

**}**

**return lang.mixin({}, {**

**fieldName: item.name,**

**label: item.alias,**

**isEditable: false,**

**tooltip: "",**

**visible: visible,**

**format: format,**

**stringFieldOption: 'textbox'**

**});**

**}));**

**var popupInfo = {**

**title: displayField ? '{' + displayField + '}' : '',**

**fieldInfos: fieldInfos,**

**description: null,**

**showAttachments: false,**

**mediaInfos: []**

**};**

**return popupInfo;**

**}**

**function buildInfoTemplate (popupInfo) {**

**var json = {**

**content: "<table>"**

**};**

**arrayUtils.forEach(popupInfo.fieldInfos, function (field) {**

**if (field.visible) {**

**json.content += "<tr><td valign='top'>" + field.label +**

**": <\/td><td valign='top'>${" + field.fieldName + "}<\/td><\/tr>";**

**}**

**});**

**json.content += "<\/table>";**

**return json;**

**}**

**function clearAll () {**

**map.graphics.clear();**

**var layerIds = map.graphicsLayerIds.slice(0);**

**layerIds = layerIds.concat(map.layerIds.slice(1));**

**arrayUtils.forEach(layerIds, function (layerId) {**

**map.removeLayer(map.getLayer(layerId));**

**});**

**}**

**function getSeparator (string) {**

**var separators = [",", " ", ";", "|"];**

**var maxSeparatorLength = 0;**

**var maxSeparatorValue = "";**

**arrayUtils.forEach(separators, function (separator) {**

**var length = string.split(separator).length;**

**if (length > maxSeparatorLength) {**

**maxSeparatorLength = length;**

**maxSeparatorValue = separator;**

**}**

**});**

**return maxSeparatorValue;**

**}**

**function zoomToData (featureLayer) {**

**// Zoom to the collective extent of the data**

**var multipoint = new Multipoint(map.spatialReference);**

**arrayUtils.forEach(featureLayer.graphics, function (graphic) {**

**var geometry = graphic.geometry;**

**if (geometry) {**

**multipoint.addPoint({**

**x: geometry.x,**

**y: geometry.y**

**});**

**}**

**});**

**if (multipoint.points.length > 0) {**

**map.setExtent(multipoint.getExtent().expand(1.25), true);**

**}**

**}**

**//File upload for older browsers**

**function uploadFile (files) {**

**if (files && files.length === 1) {**

**console.log("handle files");**

**handleCSV(files[0]);**

**}**

**else {**

**dom.byId("status").innerHTML = "Uploading…";**

**request({**

**url: "http://serverapi.arcgisonline.com/demos/csv/reflect.ashx",**

**form: dom.byId("uploadForm"),**

**load: requestSucceeded,**

**error: requestFailed**

**});**

**}**

**}**

**function requestSucceeded (response) {**

**dom.byId("status").innerHTML = "";**

**handleCSV(response);**

**}**

**function requestFailed (error) {**

**dom.byId("status").innerHTML = 'Unable to upload';**

**console.log(JSON.stringify(error));**

**}**

**}**

**);**

**</script>**

**</head>**

**<body class="claro">**

**<!--[if IE 7]>**

**<style>**

**html, body {**

**margin: 0;**

**}**

**</style>**

**<![endif]-->**

**<div id="mainWindow" data-dojo-type="dijit/layout/BorderContainer" data-dojo-props="design:'headline',gutters:false"**

**style="width:100%; height:100%;">**

**<div id="header" data-dojo-type="dijit/layout/ContentPane" data-dojo-props="region:'top'">**

**<div id="title">GroupEZ results</div>**

**</div>**

**<div data-dojo-type="dijit/layout/ContentPane" id="rightPane" data-dojo-props="region:'right'">**

**<p style="padding:4px;"><span>Mouse over your results to see more details.</span>**

**</p>**

**<form id="uploadForm" style='display:none;padding:4px;' method="post" enctype="multipart/form-data">**

**<input type="file" name="data" id="inFile" size="15" />**

**</form>**

**<span id="status"></span>**

**<div id="fileInfo">&nbsp;</div>**

**</div>**

**<div id="mapCanvas" data-dojo-type="dijit/layout/ContentPane" data-dojo-props="region:'center'">**

**<div id="clearButton" class="roundedCorner">**

**<span>Clear Map</span>**

**</div>**

**</div>**

**</div>**

**</body>**

**</html>**