**Export Web Map Specification**

**Description**
A web map is made of 5 top level objects:

```json
{  "mapOptions": {},  "operationalLayers": [ ],  "baseMap": {},  "layers": {},  "layoutOptions": { }}
```

**mapOptions**

*mapOptions* is a required object that defines map display properties.

**Syntax**

```json
{  "extent": {    "min": <min>,    "max": <max>,    "spatialReference": <spatialReference>  },  "mapMode": <mapMode>,  "spatialReference": <spatialReference>,  "time": [ <timeInstant> | <startTime>, <endTime> ]}
```

**Example**

```json
{  "extent": {    "min": -12933906.537,    "max": 3993856.886,    "spatialReference": -12933371.098,    "max": 3994375.185,    "spatialReference": 102100  },  "mapMode": 1234.5,  "rotation": -45,  "spatialReference": {    "wid": 102100  },  "time": [1199145600000, 1230768000000]}
```

**Description**

*extent*: A required property, which defines the extent of the map. The spatial reference of the extent object is optional; when it is not provided, it is assumed to be in the map’s spatial reference. When the aspect ratio of the map extent is different than the size of the map on the output page or the *exportOptions.outputSize*, you may notice more features on the output map.

*mapOptions.* The map scale at which you want your map to be exported. This property is optional, but recommended for getting expected results. The scale property is especially useful when map services in this web map have scale-dependent layers or reference scales set. Since the map that you are viewing on the web app may be smaller than the size of output map (for example 8.5 in x 11 in or A4 size), the scale of the output map will be different and you may see differences in features and/or symbols in the web application as compared with the output map. When scale is used, it takes precedence over the extent; but the output map is drawn at the requested scale centered on the center of the extent.

*rotation*: This represents the number of degrees by which the data frame will be rotated, measured counterclockwise from the north. To rotate clockwise, use a negative value.

*spatialReference*: The spatial reference of the map. The order of preference when *spatialReference* is missing is as follows:

- *mapOptions.extent.SpatialReference*
- *baseMap.baseMapLayers.SpatialReference*
- *map template spatial reference*

*time* (optional): If there is time-aware layer and you want it to be drawn at a specified time, you need to specify this property. This order list can have one or two elements. Add two elements (start time followed by end time) to represent a time extent, or provide only one element time to represent a time instant. Times are always in UTC.

**operationalLayers**

The *operationalLayers* list contains all the operational layers to be displayed in the map. The order of the array defines the order of the layers in the map. The type of each layer is defined by the URL resource response. If the resource cannot be determined from the URL, then the *type* property defines the layer type. For example, wms layer requires to specify type: “wms”.

There are some properties common to all types of operational layers, while others are specific to each type of operational layer.

**Notes**

- In case of secured layers, append a token to the url. A username and password are not supported as a part of the URL.
- For a map service, feature service, or image service, use a URL pointing to the REST endpoint of the service. SOAP endpoints are unsupported.

**Syntax**

```json
{  //ArcGIS MapService Layer     "id": "webmapOperationalLayer1",     "url": "http://myServer/myServiceLayer1",     "title": "MyServiceLayer1",     "opacity": 0.2,     "visibility": true,     "minScale": 0.0001,     "maxScale": 10000,     "visibleLayers": [ ],     "layers": [      "id": "sub layer id 1",      "layerDefinition": [    ]  ]}
```

**Example**

```json
{  //ArcGIS MapService Layer     "id": "mapServiceLayer1",     "url": "http://myServer/arcgis/rest/services/myServ
     "title": "ArcGIS MapService Layer",     "opacity": 1,     "visibility": true,     "minScale": 20000,     "maxScale": null,     "visibleLayers": [ [0,3],     "layers": [      "id": <sub layer id 1>,      "LayerDefinition": [    ]  ]
```
"minScale": "<minScale>",
"maxScale": "<maxScale>",
"extent": {
"minX": "<minX>",
"minY": "<minY>",
"maxX": "<maxX>",
"maxY": "<maxY>",
"spatialReference": "<spatialReference>",
},
"imageData": "<base64 encoded image data>"
},
// Geoprocessing Result layer
{
"id": "webmap<operationalLayerId>",
"url": "<url>({token:tokenString})",
"title": "<title>",
"opacity": "<opacity>",
"visibility": true | false,
"minScale": "<minScale>",
"maxScale": "<maxScale>"
}

Description

id (optional): A string uniquely identifying an operational layer. This is mostly needed for legends.

title (optional): Name of an operational layer. If true shows up on the legend, should the legend on layout support that.

visibility (optional): Default value is true.

opacity (optional): Value ranges from 0 to 1 (default). 0 means completely opaque, while 0 means fully transparent.

minScale (optional): Layer does not draw when zoomed out beyond this scale.

maxScale (optional): Layer does not draw when zoomed in beyond this scale.

Properties specific to map service layers:

- visibleLayers (optional): Array of sublayer IDs that should be made visible within the map service layer. If this is omitted, the operational layer is drawn with the default
- drawingInfo (optional): Use this to override the sublayer's drawing settings, for example the layer's renderer.
- renderer (optional): When specified, this overrides the sublayer's original renderer.
- source (optional): Represents the source of a layer that gets added to the map service dynamically per request.

Properties specific to feature layers:

- drawingInfo (optional): Specifies the renderer for this layer.
- definitionExpression (optional): A SQL statement that restricts which features will be drawn.
- objectIds (optional): Restrictions which features are drawn, based on object IDs.
- geometry (optional): Restrictions features to be drawn by a geometry.
- geometryType (optional): This is required when a geometry is specified.
- spatialRel (optional): The spatial relationship to be applied on the input geometry while performing the query.
- relationParam (optional): The spatial relate function that can be applied while performing the query. An example for this spatial relate function is "FFFFTTTT***"
- selectionObjectIds (optional): Highlights the features with the given object IDs with the provided symbol. selectionSymbol must be set.
- selectionSymbol (optional): Features are highlighted with this symbol. This is required when selectionObjectIds is specified.
- version (optional): Specify a geodatabase version name if you use a version other than the one referred to by the map or feature service.

Properties specific to WMS service layers:

- type (required): For WMS layers, this must be specified as "wms".
- format (optional): The requested image format from the server.
- transparentBackground (optional): When true, the background becomes transparent provided that the requested image format supports a transparent color (JPEG is an e color). The default value is false.
- visibleLayers: Array of sublayer names that should be made visible within the wms service layer. The order is important too; the order must conform to the way the WMS layers (optional): Array of sublayer names to add to the map.
- styles (optional): Use this to override a sublayer's default drawing style. When it is specified, make sure the number and order of styles match the visibleLayers array. W you can pass in an empty string, for example: "styles": ['highways', '', 'population']
- version (optional): the WMS version that you want to connect to. The default is the latest version supported by the given WMS service.

Properties specific to client-side image layers:

- type (required): For this kind of layer, the type must be "image".
- extent (required): The minimum bounding box that the image fits in.
- url (optional): URL to an image that you would like to draw. This is only required when imageData is not passed in.
- imageData (optional): The image encoded as base64. This is required when the url property is not passed in.

Adding Geoprocessing result:

There are two ways a geoprocessing result can be included,

1. When a geoprocessing result comes back as a FeatureSet and is drawn as graphics on the client side, send the result as a FeatureCollection,
2. When the result is drawn by a job's result map service, add a new map service layer in operationalLayers and set the url property to point to the endpoint of the result.
- GP result off 10.1 server: http://(myMachine:6080)/arcgis/rest/services/(GPJobMapServiceName)/MapServer/jobs/(job_id)
- GP result off pre-10.1 server: http://(myMachine)/arcgis/rest/services/(GPServiceName)/GPServer/(GPTaskName)/jobs/(job_id)/results/"out_param_name"

baseMap

The map contains one baseMap, which has title and baseMapLayers that is an ordered list of baseMapLayer. Each baseMapLayer must be in the same spatial reference and tiling scheme. When there is a baseMap, it defines the map's spatial reference.

baseMapLayers can be comprised of one or more baseMapLayer, and you can mix-and-match these.

Note: The order of a baseMapLayer inside baseMapLayers array defines the layer drawing order. The first baseMapLayer draws below the second baseMapLayer and so on, with the last baseMapLayer drawing on top of all other baseMapLayer.
```json
{
  "title": "title",
  "baseMapLayers": [
    //ArcGIS Tiled MapService Layer
    {
      "id": "<id>",
      "url": "<url>",
      "opacity": <opacity>
    },
    //Bing Map Layer
    {
      "id": "<id>",
      "type": "BingMapBasemap | BingMapAerial | BingMapHybrid",
      "culture": "<bing maps supported culture>",
      "opacity": <opacity>
    },
    //OpenStreetMap Layer
    {
      "id": "<id>",
      "type": "OpenStreetMap",
      "opacity": <opacity>
    },
    //WMS Layer
    {
      "id": "<id>",
      "type": "wms",
      "url": "<url>",
      "layer": "<layerName>",
      "style": "<style>",
      "format": "<imageFormat>",
      "tileMatrixSet": "<tileMatrixSet>",
      "opacity": <opacity>
    }
  ]
}
```

**Description**

id (optional):

opacity (optional): Set a transparency on the layer. Range of values: 0 - 1, where 0 is transparent and 1 is opaque.

**exportOptions**

`exportOptions` defines settings for the output map.

```json
{
  "dpi": <dpi>,
  "outputSize": [width, height]
}
```

**Description**

dpi (optional): The resolution in dots per inch. The default is 96 dpi.

outputSize (optional): Size of the map in pixels. The size must be defined when an empty string or MAP_ONLY (without quotes) is passed in as a value to the layoutTemplate (Export Web Map task) parameter. If `layout_template` is neither MAP_ONLY nor empty string, `layout_template` takes precedence over `outputSize`.

**layoutOptions**

`layoutOptions` defines settings for different available page layout elements, and is only needed when an available layout_template is chosen. Page layout elements include title, copyright text, scale bar, author name, and custom text elements.

This option replaces properties of existing elements only. If an element does not exist in the layout_template, the related properties get ignored. For example, if a layout_template does not have a copyright text element, layoutOptions cannot add any new copyright text element.

All the properties of this object are optional. When a value of a property is specified, the value of the corresponding page layout element is replaced; otherwise, the existing element is left untouched.

```json
{
  "titleText": "titleText",
  "authorText": "<authorName>",
  "copyrightText": "<copyright>",
  "scaleBarOptions": {
    "metricUnit": "<meters | kilometers>",
    "metricLabel": "<metricUnitLabel>",
    "nonMetricUnit": "<feet | yards | miles | nauticalMiles>",
    "nonMetricLabel": "<nonMetricUnitLabel>"
  },
  "customTextElements": {
    "<textElementName1>": "<value1>",
    "<textElementName2>": "<value2>"
  },
  "legendOptions": {
    " operationalLayers": {
      "id": "<webmapOperationalLayerId>",
      "sublayerIds": [
      ]
    },
    "titleText": "City Land Use Map",
    "authorText": "Print by: XXY",
    "copyrightText": "© esri",
    "scaleBarOptions": {
      "metricUnit": "<meters | kilometers>",
      "metricLabel": "<metricUnitLabel>",
      "nonMetricUnit": "<feet | yards | miles |
    "nonMetricLabel": "<nonMetricUnitLabel>"
  },
  "customTextElements": {
    "<townshipName": "<Town ABC>",
    "legendOptions": {
      " operationalLayers": {
        "id": "<mapserviceLayer",
        "sublayerIds": [0,1]
      },
```
```xml
<mapServerSublayerId1>,
<mapServerSublayerId2>
  
  
  
</mapServerSublayerId1>,
<mapServerSublayerId2>

```

```
  
  
  

```

**Description**

*titleText (optional):* The text of the map title text element is updated if it exists on the layout.

*authorText (optional):* The text of the author text element is updated if it exists on the layout.

*copyrightText (optional):* The text of the copyright text element is updated if it exists on the layout.

*titleBarOptions (optional):* These update a title bar if one exists.

  - *metricUnit (optional):* Sets the units of the title bar to metric units.
  - *metricLabel (optional):* String indicating how units should be labeled. Example: KM, Kilometers, or kms. When this is not specified or an empty string is passed in, the text value of the unit is used.
  - *nonMetricUnit (optional):* Sets the units of the title bar to non-metric units.
  - *nonMetricLabel (optional):* String indicating how units should be labeled. Example: Miles or mi. When this is not specified or an empty string is passed in, the text value of the unit is used.

*customTextElements (optional):* This is an array of name-value pairs. You need to use this if you want to update text of a TextElement (that is not a DynamicText) on the pagelayout. Values must be strings.

*legendOptions (optional):* Specifies properties of a legend element on the layout.

  - *operationalLayers (optional):* Specifies the operational layers whose legends will be added to the layout.
    - *id:* String representing the ID of the layer. The ID must be unique and must match the layer’s ID in the operationalLayers element where the operational layer is defined.
    - *subLayerIds (optional):* An array of sublayers whose element types are string or long and contextual to the operational layer type. For example, for a map service layer, it must be a number; for WMS layer, it must be string. subLayerIds are recommended to be specified for operational layers that have sublayers. Once omitted, in the case of map service and WMS layers, legends from all sublayers are added to the legend element on the layout. For feature or graphics layers, the sublayers property does not need to be set.