



aptomar

DRAFT for Comment

Document Title:	Doc Rev:	Document No:	Doc. Responsible:
aptomar file format description	1	:	MTL



## Contents

1 Aptomar asset bundle format .....	2
1.1 Goals .....	2
1.2 Format .....	2
1.2.1 Archive organization .....	2
1.2.2 <i>manifest.json</i> .....	3
1.3 <i>manifest.json</i> entries .....	3
1.3.1 Including Files .....	3
1.3.2 Field summary .....	4
1.3.3 Field details .....	5
1.4 Examples .....	11
1.4.1 Map Data Layers .....	11
1.4.2 Image .....	13
1.4.3 Video .....	13
1.5 Useful Information .....	14
1.6 Open Questions .....	14
2 Appendix: <i>manifest.json</i> Schema .....	14

## 1 Aptomar asset bundle format

This document describes the **draft** description of the data file that describes customer assets for import into the aptomar system.

This is a **draft** and is open for discussion.

### 1.1 Goals

The goal of this file format is to describe a method for generic import and export of data from an Aptomar system. This document describes a general format and provides a mechanism for extending the format for new data types. This document also describes several examples of common uses for this format.

### 1.2 Format

The file is a zip-formatted archive containing the following:

- All files related to the asset
- A *manifest.json* file describing the files and organization in the archive
- The archive should have the extension *.apt*.

#### 1.2.1 Archive organization

A hypothetical archive could look as follows:



DRAFT for Comment

```
<archive>.apt
|-- <folder>
|   |-- file1.xxx
|   |-- file2.yyy
|   `-- file3.zzz
|-- manifest.json
`-- resources
    |-- img.png
    `-- img.jpg
```

The archive *may* be but is not *required* to be compressed with the *deflate* zip compression. Depending on the data included in the archive, compression may be either superfluous or counter-productive. In general, it is recommended to begin with an uncompressed archive.

### 1.2.2 *manifest.json*

The *manifest.json* file describes the layout and structure of the archive. It has the following format for the contents:

```
{
  "key-1": "value-1",
  "key-2": "value-2",
  ...
  "key-n": "value-n"
}
```

That is, the manifest is a valid JSON-formatted file that contains *key-value* tuples that describe the data in the archive.

## 1.3 *manifest.json* entries

This section describes the keys and values that are expected and required inside the manifest file. The keys break down into common keys that can be in any manifest and data-type-specific keys that apply to files with only certain types of data.

This section is intended to be descriptive **only**. *manifest.json* files **must** validate against the current schema to be considered correct.

### 1.3.1 Including Files

The main purpose of a *manifest.json* file is to describe the data included in and needed by an archive file. Most data references in the manifest are of the following form:

```
"some-key": {
  "data": ["file:/foo.txt", "data:text/x-sld,<data>",
    "http://foo.bar.com/foo.jpg"],
  "type": "text/x-ewkt"
}
```



The two required keys are “data” and “type”. The “type” is a pseudo-mime type that describes the data. The types that are accepted depend on the context and are described below. These are typically something like the following: “text/x-sld”, “text/x-worldfile”, “image/jpeg” or the like. See the descriptions below for the supported content types.

The “data” field is an array of files that encompass the data. It is expected in most cases that this will be an array with one element, but some particular types of data may have multiple parts. For example, an ESRI shapefile includes at least three components.

A “data” element may specify data in one of three ways:

1. “data”: [“file:/foo.txt”] indicates that the data may be found in a file “foo.txt” included in the root of the archive.
2. “data”: [“data:,Data as a string”] indicates that data is included as an encoded URI. This follows the normal rules of data URIs (e.g. RFC 2397) and can include content such as images directly in the manifest.
3. “data”: [“http://foo.com/foo.txt”] indicates that the file can be retrieved at the specified URI. This is somewhat dangerous as there is no guarantee as to *when* the client will retrieve the data.

### 1.3.2 Field summary

```
{
  // Required
  "date": "2012-12-19T08:46Z",
  "description": "Useful description of the contents",
  "generator": {...},
  "manifest_version": 1,    // Currently version 1

  // One of the following

  // Map data layer fields
  "asset": {...},

  // Image fields
  "image": {...},

  // Video fields
  "video": {...},

  // User-created data fields (pick one)
  "point": {...},
  "route": {...},
  "area": {...},

  // observation annotations
  "observation": {...}

  // sensor metadata
  "sensor": {...}
}
```



aptomar

DRAFT for Comment

### 1.3.3 Field details

#### 1.3.3.1 date

This key describes the date and time the archive was generated in ISO 8601 format.

*Example:* "2012-12-19T08:46Z" means the archive was created at 8:46 UTC on December 19, 2012.

#### 1.3.3.2 description

The description is a key that contains an informational string that can be displayed to the user. It should contain some relevant information about the file.

*Example:* "Winter 2012 extraction of asset data"

#### 1.3.3.3 generator

The generator is a dictionary that contains additional information that can be displayed to the user.

```
{
  "program": "Name / ID of the generator program",
  "creator": "Contact string for the creator of the file"
}
```

#### 1.3.3.4 manifest\_version

This describes the version of the manifest file that is needed to parse the file. Required for future compatibility.

#### 1.3.3.5 asset

Assets are dictionaries containing data layers and grouping information. These are used to import customer-specific vector map data.

##### 1.3.3.5.1 layers

Layers are an dictionary of dictionaries where each entry describes one map layer in the archive. Each dictionary entry should contain the following:

```
"layer-key-1": {
  "name": "Layer 1 Name",
  "geometry": {...},
  "style": {...},
  "resources": {...}
}
```



aptomar

DRAFT for Comment

The “key” (“layer-key-1” in the example) is an internal reference used to refer to the layer. It must be unique. “name” is a human-readable description of the layer.

“geometry”, “style” and “resources” are all data references. The only currently supported formats are the following:

key	format
geometry	“text/x-shapefile”
style	“text/x-sld”
resources	varies depending on the resource.
	Most “image/*” formats supported.

### 1.3.3.5.2 groups

Groups describe the relative organization of layers into groups of easier display. For clarity, groups should be limited to a relatively small number (<10) and layers **must** be described in at least one group or they will not be displayed. In general, expect that only one group may be displayed at any time.

```
"group-1": {
  "name": "Group 1 Name",
  "layers": ["layer-key-1", "layer-key-2"]
}
```

The “key” (“group-1” in the example) is an internal reference used to refer to the group. It must be unique. “name” is a human-readable description of the group.

“layers” is an array of references to layers. Each element of the array is a key in the root layers dictionary.

### 1.3.3.6 image

The image field describes a single image contained in an archive.

```
{
  "name": "The image name",
  "description": "A detailed description of the image",
  "created": "2012-12-19T08:46Z",

  "data": {...},
  // The following are optional
  "georeference": {...},
  "bounds": {...}
}
```

“name” is a short title for the image. More information can be provided in the “description” field. “data” is a file reference to the image file data.



aptomar

DRAFT for Comment

The “georeference” is the location of the sensor when the image was captured. The georeference is a dictionary with the following keys:

```
{
  // WGS-84
  "longitude": 0.000,
  "latitude": 0.000,
  "elevation": 0.000 // in meters
}
```

The “bounds” represent how to map the image data into the map. The method for processing the bounds depends on the “bound-type”. The format for a “bounds” object is the following:

```
{
  "data": ["..."],
  "type": "text/x-worldfile"
}
```

The “type” specifies the format for the bounds option. Presently the only acceptable format is “text/x-worldfile”, corresponding to the [ESRI World file format](#). “data” is described in the section about file references above.

An image “bounds” may also be represented in the following structured format:

```
"bounds": {
  "pixel_width": 0.000415,
  "pixel_height": -0.000141,
  "pixel_units": "degrees",
  "rotation_x": 0,
  "rotation_y": 0,
  "upperleft": {
    "latitude": 70.014783,
    "longitude": 15.127447
  }
}
```

### 1.3.3.7 video

The video field describes a single video contained in an archive.

```
{
  "name": "The video name",
  "description": "A detailed description of the video",
  "created": "2012-12-19T08:46Z",

  // Videos can have a georeference as well
}
```



aptomar

DRAFT for Comment

```
"georeference": {...},

// One of the following
"data": {...}
}
```

“name” is a short title for the video. More information can be provided in the “description” field. “created” contains the ISO 8601 date corresponding to when the video was created.

“data” refers to the filename of the video inside the archive. However, since video files can be quite large, it may be useful to simply encode a reference to the video as a URI. The reader of the file is then responsible for retrieving the actual video data. See the section about file references above.



### 1.3.3.8 point

The point field describes a single point-of-interest contained in an archive.

```
{
  "name": "The point name",
  "description": "A detailed description of the point",
  "created": "2012-12-19T08:46Z",
  "object-type": "...",
  "geometry": {...}
}
```

“name” is a short title for the point-of-interest. More information can be provided in the “description” field. “created” contains the ISO 8601 date corresponding to when the point-of-interest was created.

“object-type” describes the type of point and must be in the set of points that Aptomar systems can parse. The following table lists the supported types and meanings.







type	icon
boat	
bouy	
debris	
fishfarm	
green	





aptomar

DRAFT for Comment

type	icon
oil	
personnel	
red	
unknown	
vessel	
yellow	

“geometry” is an object with the following format:

```
{
  "type": "text/x-ewkt",
  "data": ["..."]
}
```

In this object, “data” is either an array of text strings containing the geometry or a reference to a file containing the geometry. Interpretation is controlled by the “type” field. The only supported geometry type at present is “text/x-ewkt”. “ewkt” is the “Extended Well-Known Text” representation of the location of the point-of-interest. This must use the “POINT” WKT type.

### 1.3.3.9 route

The route field describes a single route contained in an archive.

```
{
  "name": "The route name",
  "description": "A detailed description of the route",
  "created": "2012-12-19T08:46Z",
  "geometry": {...}
}
```

“name” is a short title for the point-of-interest. More information can be provided in the “description” field. “created” contains the ISO 8601 date corresponding to when the point-of-interest was created.

“geometry” is an object with the following format:



aptomar

DRAFT for Comment

```
{
  "type": "text/x-ewkt",
  "data": ["..."]
}
```

In this object, “data” is either an array of text strings containing the geometry or a reference to a file containing the geometry. Interpretation is controlled by the “type” field. The only supported geometry type at present is “text/x-ewkt”. “ewkt” is the “Extended Well-Known Text” representation of the location of the point-of-interest. This must use the “LINESTRING” WKT type.

### 1.3.3.10 area

The area field describes a single area contained in an archive.

```
{
  "name": "The area name",
  "description": "A detailed description of the route",
  "created": "2012-12-19T08:46Z",
  "geometry": {...}
}
```

“name” is a short title for the point-of-interest. More information can be provided in the “description” field. “created” contains the ISO 8601 date corresponding to when the point-of-interest was created.

“geometry” is an object with the following format:

```
{
  "type": "text/x-ewkt",
  "data": ["..."]
}
```

In this object, “data” is either an array of text strings containing the geometry or a reference to a file containing the geometry. Interpretation is controlled by the “type” field. The only supported geometry type at present is “text/x-ewkt”. “ewkt” is the “Extended Well-Known Text” representation of the location of the point-of-interest. This must use the “POLYGON” WKT type.

### 1.3.3.11 observation

An observation describes an interesting region in another image or geometry. It is defined as follows:

```
"observation": {
  "date": "2013-05-27T08:46:00Z",
  "bounds_nw": {
    "longitude": 15.139300,
    "latitude": 70.011900
  }
}
```



aptomar

DRAFT for Comment

```
    },  
    "bounds_se": {  
      "longitude": 15.334700,  
      "latitude": 69.919100  
    }  
  }  
},
```

This key is currently only used in conjunction with georeferenced image and sensor keys. It is otherwise ignored.

### 1.3.3.12 sensor

A “sensor” dictionary provides information about the sensor that generated a reading. This key is **required** when importing images with georeferences. The format for this key is as follows:

```
"sensor": {  
  "shortname": "Sensor 1",  
  "location_name": "My Vessel",  
  "manufacturer_name": "Company A/S",  
  "model_number": "Reading Generator 1.2",  
  "serial_number": "1234567"  
}
```

All keys are required.

## 1.4 Examples

### 1.4.1 Map Data Layers

This section shows an example of how to encapsulate both geometry and styling information for any number of map data layers in a single file. The file also contains a description of the enclosed layers and describes how they are intended to be organized and displayed.

The file is a zip-formatted archive with *.apt* extension containing the following:

- layer data in ESRI [shapefile format](#)
- styling information for each layer in [Styled Layer Descriptor format](#)
- all additional resources needed by the layer or styling data

A hypothetical archive could look as follows:

```
geom.apt  
|-- layers  
|   |-- layer1.dbf  
|   |-- layer1.shp  
|   `-- layer1.shx
```



# aptomar

## DRAFT for Comment

```
|-- manifest.json
|-- resources
|-- styles
   |-- layer1.sld
```

And the corresponding manifest file could look like the following:

```
{
  "date": "2012-12-19T08:46Z",
  "description": "Text description of the contents for ↵
presentation in a confirmation dialog.",
  "generator": {
    "program": "SuperCool.app",
    "creator": "Aptomar AS"
  },
  "manifest_version": 1,
  "asset": {
    "layers": {
      "layer-1": {
        "name": "Layer 1 Name",
        "geometry": {
          "type": "shapefile",
          "data": ["file:/layers/layer1.shp", ↵
          "file:/layers/layer1.dbf", ↵
          "file:/layers/layer1.shx"]
        },
        "style": {
          "data": ["file:/styles/layer1.xml"],
          "type": "sld"
        },
        "resources": {
          "data": ["file:/resource1.png", ↵
          "file:/resource2.png"],
          "type": "image/png"
        }
      },
      "groups": {
        "group-1": {
          "name": "Group 1 Name",
          "layers": ["layer-1"]
        }
      }
    }
  }
}
```



DRAFT for Comment

### 1.4.2 Image

A hypothetical archive could look as follows:

```
geotagged-image.apt
|-- manifest.json
`-- image.jpg
```

And the corresponding manifest file could look like the following:

```
{
  "date": "2012-12-19T08:46Z",
  "description": "Text description of the contents for ↵
presentation in a confirmation dialog.",
  "generator": {
    "program": "SuperCool.app",
    "creator": "Aptomar AS"
  },
  "manifest_version": 1,
  "image": {
    "name": "The image name",
    "description": "A detailed description of the image",
    "created": "2012-12-19T08:46Z",
    "data": ["file:image.jpg"],
    "georeference": {
      "longitude": 10.4344,
      "latitude": 63.4181,
      "elevation": 150.60
    },
    "bounds": {
      "type": "text/x-worldfile",
      "data": ["data:,bounds as a string, interpreted by type"]
    }
  }
}
```

### 1.4.3 Video

A hypothetical archive could look as follows:

```
video.apt
|-- manifest.json
`-- clip.mp4
```

And the corresponding manifest file could look like the following:



aptomar

DRAFT for Comment

```
{
  "date": "2012-12-19T08:46Z",
  "description": "Text description of the contents for ↵
presentation in a confirmation dialog.",
  "generator": {
    "program": "SuperCool.app",
    "creator": "Aptomar AS"
  },
  "manifest_version": 1,
  "video": {
    "name": "The video name",
    "description": "A detailed description of the video",
    "created": "2012-12-19T08:46Z",
    "georeference": {
      "longitude": 10.4344,
      "latitude": 63.4181,
      "elevation": 150.60
    },
    "data": ["clip.mp4"]
  }
}
```

## 1.5 Useful Information

The [Well-Known Text](#) format is a simple way to textually describe points, lines, and polygons.

These manifest files are written using [Javascript Object Notation](#), a simple and commonly used method for data interchange.

The [ESRI World file](#) is a plain text file that is commonly used to georeference images.

## 1.6 Open Questions

1. Should we have a “uuid” field for the data elements to allow for updating?
2. Should we support WKB in addition to WKT?
3. What about other formats like KML?
4. Should we support a Zip format for the archive? What extension?

## 2 Appendix: *manifest.json* Schema

The electronic version may be obtained from <https://raw.githubusercontent.com/aptomar/apt-file-format/master/src/apt.schema.json>.