ESRI Imagery Compact Cache V2 Technical Description

An ESRI White Paper

Compact Cache V2 is an Esri tile storage format introduced with the ArcGIS 10.2 release, offering improved performance over the original Compact Cache. The purpose of this white paper is to document the internal structure of the Compact Cache V2 as used for map and imagery storage, to allow programmers to read and write data in Compact Cache V2.

Contents

[Why Compact Cache V2? 1](#_Toc468276515)

[Disk Structure 2](#_Toc468276516)

[Configuration File 3](#_Toc468276517)

[Bundle File Names 4](#_Toc468276518)

[Bundle File Structure 4](#_Toc468276519)

[Bundle Header 5](#_Toc468276520)

[Tile Index 5](#_Toc468276521)

[Tile Index Record 6](#_Toc468276522)

[Tile Record 6](#_Toc468276523)

[Appendix A: Configuration File Content Example 7](#_Toc468276524)

# Why Compact Cache V2?

Storing pre-rendered map tiles for later use is a well-established method of improving GIS system performance. The Esri Exploded Cache implementation stored each such tile as an individual file in a specific folder, using a simple naming convention to map a specific tile to a given map location and resolution. While efficient and simple, this approach relies heavily on the file system, which leads to poor performance, especially for very large caches. Since many map tiles are much smaller than the allocation size on a modern storage system, Exploded Caches also tend to waste storage space.

As a solution to the Exploded Cache scalability problems, the original Esri Compact Cache stored all the tiles for a larger map area in a single file called a bundle, file which follows a similar naming convention to the one used for Exploded Cache. Compact Cache bundle files store an area of 128 by 128 tiles. Since most map access focuses on a specific location at a time, only one or a couple of bundle files are in use, greatly reducing the file system load and improving performance. To locate a given tile within the larger bundle file, a spatial index table is used. In the original Compact Cache, this index held only the offsets for all the tiles within a bundle. The index file was stored independently from the tile bundle, with a matching file name but different extension.

The Compact Cache V2 is similar to the original Compact Cache, with two primary differences: the Compact Cache V2 index table contains both the offset and size of each tile, and the index table is stored within the bundle file. These two changes further improve the application performance, reducing the number of input/output operations needed to access the tiles. While originally used only to store image tiles in JPEG or PNG format, the Compact Cache V2 structure lends itself to other uses, for example storing terrain data or vector tiles. This document will mostly be concerned with the naming and structure of the Compact Cache V2 bundle files when used to store map images. The internal tile format or the exact cache metadata format may vary with the use, and indeed are the current subject of technology development within Esri.

Figure 1. Compact Cache V2 Disk Organization

R0000C0080.bundle

CacheFolder/

conf.xml

conf.cdi

\_alllayers/

L08/

# Disk Structure

The Compact Cache V2 disk structure (illustrated in Figure 1) consists of a main folder containing a configuration file named *conf.xml*, an envelope file named *conf.cdi*, and a *\_alllayers* folder, which itself contains level subfolders. Each level folder contains all the bundles with tiles rendered for a specific scale or level of detail (LOD).  
The *conf.xml* file contains most metadata properties of the cache in XML format. These properties include identification of the cache as a Compact Cache, spatial reference, top-left origin, tile size in pixels, tile image format, target screen resolution, the set of predefined LOD scales at which the tiles exist, and the size of the bundle measured in tiles. Each scale defined in the *conf.xml* represents an LOD level, counted up from zero in order of decreasing scale. Two successive level scales differ by a factor of two. The content of the *conf.xml* is sufficient to determine the tile grid used by the cache and establish the mapping between the map coordinates and tiles within specific bundles. Every tile within a cache is uniquely identified by the level, row and column numbers. The row and column are measured from the top left origin of the cache, which itself is defined in the *conf.xml* file.  
The extent file *conf.cdi* contains the envelope of the extent of data in the cache, also in XML format. It is used to eliminate slow file search operations for non-existing tile, mostly an Exploded Cache problem.

# Configuration File

A Compact Cache V2 configuration file called *conf.xml* resides in the main cache folder (See Appendix A for an example). This XML file describes various properties of the cache. There are three main objects: TileCacheInfo, TileImageInfo and CacheStorageInfo.

The TileCacheInfo object contains metadata properties of the cache as a whole. The principal fields of a TileCacheInfo object are described in Table 1

Table 1. TileCacheInfo

|  |  |
| --- | --- |
| **Element** | **Description** |
| SpatialReference | Spatial reference of the cache |
| TileOrigin | X and Y coordinates of the top left corner of the top left tile in the cache spatial reference system |
| TileRows | Tile width in pixels. Usually 256 or 512 |
| TileHeight | Tile height in pixels. Usually 256 or 512 |
| DPI | The DPI (dots per inch) value used to render the tiles |
| LODInfos | A list of LODInfo elements, each one containing the level number, pixel resolution in map units, and the map scale |

The resolution field in the list of LODInfos elements provides the most accurate conversion between the pixel location and map coordinates. The corresponding scale value should agree with the resolution and DPI, but is advisory only and might not have full accuracy. The DPI value can also be used to scale the cache tiles to the actual screen resolution to ensure readability of labels and other features.

The TileImageInfo object describes the tile encoding used in the cache. The principal elements of a TileImageInfo object are described in Table 2

Table . TileImageInfo

|  |  |
| --- | --- |
| **Element** | **Description** |
| CacheTileFormat | The internal format of the tiles. Possible values for imagery are:  JPEG, PNG8, PNG24, PNG32, PNG, MIXED |
| CompressionQuality | JPEG Q setting used to generate the tiles, if the tiles are JPEG |
| Antialiasing | True if antialiasing was used during cache generation |

The CacheStorageInfo object defines the storage format and parameters of bundles. The principal elements of a CacheStorageInfo object are described in Table 3

Table 3 CacheStorageInfo

|  |  |
| --- | --- |
| **Element** | **Description** |
| StorageFormat | esriMapCacheStorageModeCompactV2 |
| PacketSize | 128, meaning that each bundle file covers an area of 128 by 128 tiles |

# Bundle File Names

In the main cache folder, in addition to the *conf.xml* and *conf.cdi* files, there is a subfolder named \_*alllayers*. Within this folder are subfolders matching each of the cache LOD. The folder names for the LOD levels begin with the uppercase letter L, followed by the level number as two decimal digits, prefixed by zero if necessary (e.g. L08). All the bundle files for a given level are stored in the respective level folder. Each bundle file stores the tiles for a 128 by 128 tile region of the cache, where 128 is the value of the PacketSize field of the CacheStorageInfo object stored in the cache configuration file.

The name of each bundle file is of the form R<rrrr>C<cccc>.bundle. The <rrrr> and <cccc> fields are the row and column number of the top- and left-most tile location covered by the bundle, in hexadecimal. The row and column fields in the bundle file name have at least four hexadecimal digits, prefixed by 0 if necessary, using lower case. Use of lower case hex digits ensures that the uppercase C character used as the separator between the row and column digits can be distinguished from the hexadecimal c. The row and column fields in the bundle name are always multiples of the PacketSize, which means that the row and column fields of the bundle file names will always end in 00 or 80.

# Bundle File Structure

Compact Cache uses bundle files for storing image tiles. A bundle file has a small header, followed by the tile index table, followed by the tile data. A bundle stores all the existing tiles for a geographical 128-by-128-tile area, for a maximum of 16,384 individual tiles. Internal to the bundle, the tiles are indexed by row and column. Each tile index record consists in tile size and tile offset within the bundle file. The index records for all 16,384 tiles within the bundle are stored in a row major array. A tile index with a size of zero means that the tile does not exist, regardless of the offset value.

# Bundle Header

The bundle header is a 64-byte structure located at the beginning of the bundle file. It consists of unsigned integers of 4 or 8 bytes, in lower-endian representation, as shown in Table 4.

Table . Bundle header.

|  |  |  |  |
| --- | --- | --- | --- |
| **File Offset** | **Bytes** | **Field** | **Value** |
| 0 | 4 | Version | 3 |
| 4 | 4 | Record Count | 16384 |
| 8 | 4 | Maximum Tile Size | Variable |
| 12 | 4 | Offset Byte Count | 5 |
| 16 | 8 | Slack Space | Variable |
| 24 | 8 | File Size | Variable |
| 32 | 8 | User Header Offset | 40 |
| 40 | 4 | User Header Size | 20 + 131072 |
| 44 | 4 | Legacy | 3 |
| 48 | 4 | Legacy | 16 |
| 52 | 4 | Legacy | 16384 |
| 56 | 4 | Legacy | 5 |
| 60 | 4 | Index Size | 131072 |

The Slack Space value is a rough measure of the amount of bytes that are part of the bundle but are not used. It is used and updated by the ArcGIS cache editing tools. The value contained here is not always accurate and should be left as zero. The Legacy fields are there to ease the transition from the earlier version of Compact Cache, they should always have the default values.

# Tile Index

The tile index immediately follows the bundle header, at file offset 64. The index is a 128-by-128 array of tile index records, each 8 bytes long, one such per tile. The total tile index size is 131,072 bytes. This value is stored at file offset 60 within the bundle header. The index records are stored in row major order in top-left orientation. This means that the first tile index record corresponds to the top left tile within the bundle, whose absolute row and column are also used to name the bundle file. The next index record is located immediately to the right of the first one. The formula to compute the offset of the index record for a specific tile is:

TileIndexOffset = 64 + 8 \* (128 \* row + column)

where row and column numbers are relative to the row and column of the top-leftmost tile of the bundle. Alternatively, a formula using the modulo operation % and the absolute row and column numbers relative to the top-leftmost tile within a level can be used:

TileIndexOffset = 64 + 8 \* (128 \* (row % 128) + (column % 128))

# Tile Index Record

To find the content of a tile located at a specific row and tile location, first locate and read the tile index record matching the row and location as described above. The tile offset and size can then be extracted from the tile index record and used to read the tile content itself. Each tile data is contigous, but the order of the tiles within the bundle file is not defined. It is possible to have unused space between tiles in the bundle file.  
A tile index record is an 8-byte, low-endian unsigned integer which contains both the starting tile offset within the file and the size of a tile. The tile offset is stored in bits 0 to 39, while the tile size is located in bits 40 to 63. Bit 0 is the least significant bit, and both offset and size are stored in little-endian format.  
If IDX is the tile index value for a particular tile, and M is 2 to the power of 40, then the offset and size can be calculated as:

TileOffset = IDX % M

TileSize = Floor(IDX / M)

All values used in the calculation above have to be 8-byte-or-larger integers. The TileOffset is an absolute offset within the bundle file indicating the location where the tile data starts. The TileSize is the size of the tile in bytes. If a decoded index has a TileSize of zero, then the tile is not stored in the bundle, regardless of the TileOffset value.

# Tile Record

The original Compact Cache bundles stored the tile data prefixed by a 4 byte little endian integer field which stored the size of the tile. To ease the transition to Compact Cache V2, this size field is retained. Note that the file offset stored in the tile index record is the offset of the first byte of the tile, not the offset of the size record, which will start at offset -4.

# Appendix A: Configuration File Content Example

<?xml version="1.0" encoding="utf-8" ?>

<CacheInfo xsi:type='typens:CacheInfo' xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance' xmlns:xs='http://www.w3.org/2001/XMLSchema' xmlns:typens='http://www.esri.com/schemas/ArcGIS/10.0'>

<TileCacheInfo xsi:type='typens:TileCacheInfo'>

<SpatialReference xsi:type="typens:GeographicCoordinateSystem">

<WKT>GEOGCS["GCS\_WGS\_1984",DATUM["D\_WGS\_1984",SPHEROID["WGS\_1984",6378137.0,298.257223563]],PRIMEM["Greenwich",0.0],UNIT["Degree",0.0174532925199433]]</WKT>

<XOrigin>-399.99999999999989</XOrigin>

<YOrigin>-399.99999999999989</YOrigin>

<XYScale>11258999068426.24</XYScale>

<ZOrigin>-100000</ZOrigin>

<ZScale>10000</ZScale>

<MOrigin>-100000</MOrigin>

<MScale>10000</MScale>

<XYTolerance>8.9831528411952133e-009</XYTolerance>

<ZTolerance>0.001</ZTolerance>

<MTolerance>0.001</MTolerance>

<HighPrecision>true</HighPrecision>

<LeftLongitude>-180</LeftLongitude>

<WKID>4326</WKID>

</SpatialReference>

<TileOrigin xsi:type='typens:PointN'>

<X>-180.0</X>

<Y>90.0</Y>

</TileOrigin>

<TileCols>256</TileCols>

<TileRows>256</TileRows>

<DPI>96</DPI>

<PreciseDPI>96</PreciseDPI>

<LODInfos xsi:type="typens:ArrayOfLODInfo">

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>0</LevelID>

<Scale>295828763.79585470937713011037</Scale>

<Resolution>0.703125</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>1</LevelID>

<Scale>147914381.89792735468856505518</Scale>

<Resolution>0.3515625</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>2</LevelID>

<Scale>73957190.948963677344282527592</Scale>

<Resolution>0.17578125</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>3</LevelID>

<Scale>36978595.474481838672141263796</Scale>

<Resolution>0.087890625</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>4</LevelID>

<Scale>18489297.737240919336070631898</Scale>

<Resolution>0.0439453125</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>5</LevelID>

<Scale>9244648.868620459668035315949</Scale>

<Resolution>0.02197265625</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>6</LevelID>

<Scale>4622324.4343102298340176579745</Scale>

<Resolution>0.010986328125</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>7</LevelID>

<Scale>2311162.2171551149170088289872</Scale>

<Resolution>0.0054931640625</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>8</LevelID>

<Scale>1155581.1085775574585044144937</Scale>

<Resolution>0.00274658203125</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>9</LevelID>

<Scale>577790.55428877872925220724681</Scale>

<Resolution>0.001373291015625</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>10</LevelID>

<Scale>288895.27714438936462610362340</Scale>

<Resolution>0.0006866455078125</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>11</LevelID>

<Scale>144447.63857219468231305181171</Scale>

<Resolution>0.00034332275390625</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>12</LevelID>

<Scale>72223.819286097341156525905853</Scale>

<Resolution>0.000171661376953125</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>13</LevelID>

<Scale>36111.909643048670578262952926</Scale>

<Resolution>0.0000858306884765625</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>14</LevelID>

<Scale>18055.954821524335289131476463</Scale>

<Resolution>0.00004291534423828125</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>15</LevelID>

<Scale>9027.977410762167644565738231</Scale>

<Resolution>0.000021457672119140625</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>16</LevelID>

<Scale>4513.9887053810838222828691158</Scale>

<Resolution>0.0000107288360595703125</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>17</LevelID>

<Scale>2256.9943526905419111414345579</Scale>

<Resolution>0.00000536441802978515625</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>18</LevelID>

<Scale>1128.4971763452709555707172788</Scale>

<Resolution>0.000002682209014892578125</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>19</LevelID>

<Scale>564.24858817263547778535863938</Scale>

<Resolution>0.0000013411045074462890625</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>20</LevelID>

<Scale>282.12429408631773889267931988</Scale>

<Resolution>0.00000067055225372314453125</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>21</LevelID>

<Scale>141.06214704315886944633965975</Scale>

<Resolution>0.000000335276126861572265625</Resolution>

</LODInfo>

<LODInfo xsi:type='typens:LODInfo'>

<LevelID>22</LevelID>

<Scale>70.531073521579434723169829875</Scale>

<Resolution>0.0000001676380634307861328125</Resolution>

</LODInfo>

</LODInfos>

</TileCacheInfo>

<TileImageInfo xsi:type='typens:TileImageInfo'>

<CacheTileFormat>JPEG</CacheTileFormat>

<CompressionQuality>75</CompressionQuality>

<Antialiasing>false</Antialiasing>

</TileImageInfo>

<CacheStorageInfo xsi:type="typens:CacheStorageInfo">

<StorageFormat>esriMapCacheStorageModeCompactV2</StorageFormat>

<PacketSize>128</PacketSize>

</CacheStorageInfo>

</CacheInfo>