GUIDELINES FOR HANDLING IMAGE METADATA

Version 1.0.1

February 2009

METADATA WORKING GROUP

www.metadataworkinggroup.org

Copyrights

Terms and Conditions

This document is made available by Adobe Systems Inc., Apple Inc., Canon Inc., Microsoft Corp., Nokia Corp. and Sony Corp. (collectively, the "Authors") and grants you (either an individual or an entity) and your affiliates ("Licensee") this license. Licensee agrees that Licensee has read, understood and will comply with these terms and conditions.

1. Definitions

1.1 "*Licensed Products*" means only those specific portions of products (hardware, software or combinations thereof) that implement and are compliant with all Normative Portions of the Guidelines.

1.2 "*Normative Portions*" means a portion of the Guidelines that must be implemented to comply with such guidelines. If such guidelines define optional parts, Normative Portions include those portions of the optional part that must be implemented if the implementation is to comply with such optional part.

1.3 "*Necessary Claims*" are those claims of a patent or patent application, throughout the world, excluding design patents and design registrations, owned or controlled, or that can be sublicensed in compliance with the requirements of this Agreement, by the party or its affiliates now or at any future time and which would necessarily be infringed by implementation of the Guidelines. A claim is necessarily infringed hereunder only when it is not possible to avoid infringing it because there is no non-infringing alternative for implementing the Normative Portions of the Guidelines. Notwithstanding the foregoing, Necessary Claims shall not include any claims other than as set forth above even if contained in the same patent as Necessary Claims; or that read solely on any implementations of any portion of the Guidelines that are not required by the Normative Portions of the Guidelines, or that, if licensed, would require a payment of royalties by the licensor to unaffiliated third parties. Moreover, Necessary Claims shall not include (i) any enabling technologies that may be necessary to make or use any Licensed Product but are not themselves expressly set forth in the Guidelines (e.g., semiconductor manufacturing technology, compiler technology, object oriented technology, basic operating system technology, data and voice networking technology, and the like); or (ii) the implementation of other published standards developed elsewhere and merely referred to in the body of the Guidelines, or (iii) any Licensed Product and any combinations thereof the purpose or function of which is not required for compliance with the Guidelines. For purposes of this definition, the Guidelines shall be deemed to include only architectural and interconnection requirements essential for interoperability and shall not include any implementation examples unless such implementation examples are expressly identified as being required for compliance with the Guidelines.

1.4 "*Guidelines*" mean the Guidelines For Handling Image Metadata Version 1.0.

Normative Sections

This document attempts to conform to the keyword usage practices defined in RFC 2119. This RFC defines the use and strength of the capitalized terms MUST, MUST NOT, SHOULD, SHOULD NOT and MAY. All sections and appendixes, except the first chapter “Introduction”, are normative, unless they are explicitly indicated to be informative.

These imperatives are used to highlight those requirements that are required to insure interoperability and drive compatibility.

References

This document includes the following references to third party documents:

Metadata Specifications

Exif 2.21/DCF 2.0

Official Exif specifications are available in paper form and can be ordered on the JEITA website.

IPTC-IIM 4.1

http://www.iptc.org/std/IIM/4.1/specification/IIMV4.1.pdf

IPTC Core 1.0

http://www.iptc.org/std/Iptc4xmpCore/1.0/specification/Iptc4xmpCore_1.0-spec-XMPSchema_8.pdf

IPTC Core 1.1 & IPTC Extension 1.0

http://www.iptc.org/std/photometadata/2008/specification/IPTC-PhotoMetadata-2008_1.pdf

XMP

http://www.adobe.com/devnet/xmp/pdfs/xmp_specification.pdf

File Format Specifications

JPEG

http://www.jpeg.org/jpeg/

TIFF

http://partners.adobe.com/public/developer/en/tiff/TIFF6.pdf

PSD/PSIRs

http://www.adobe.com/go/psir

Miscellaneous

RDF

http://www.w3.org/TR/rdf-schema

Dublin Core

http://dublincore.org/documents/dces

RFC2119

http://www.ietf.org/rfc/rfc2119.txt

Date and Time (W3C)

http://www.w3.org/TR/NOTE-datetime

TABLE OF CONTENTS

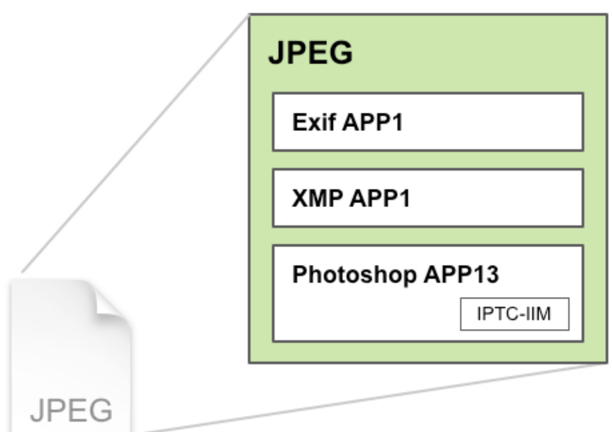
1. INTRODUCTION

Metadata, often referred to as “data about data,” provides interesting information that supplements the primary content of digital documents. Metadata has become a powerful tool to organize and search through the growing libraries of image, audio and video content that users are producing and consuming. This is especially important in the area of digital photography where, despite the increased quality and quantity of sensor elements, it is not currently practical to organize and query images based only on the millions of image pixels. Instead, it is best to use metadata properties that describe what the photo represents and where, when and how the image was taken.

Metadata is now critical in workflows ranging from consumer sharing experiences to professional-level asset management. That said, there are several complications which result from structural hierarchies required to store metadata within images:

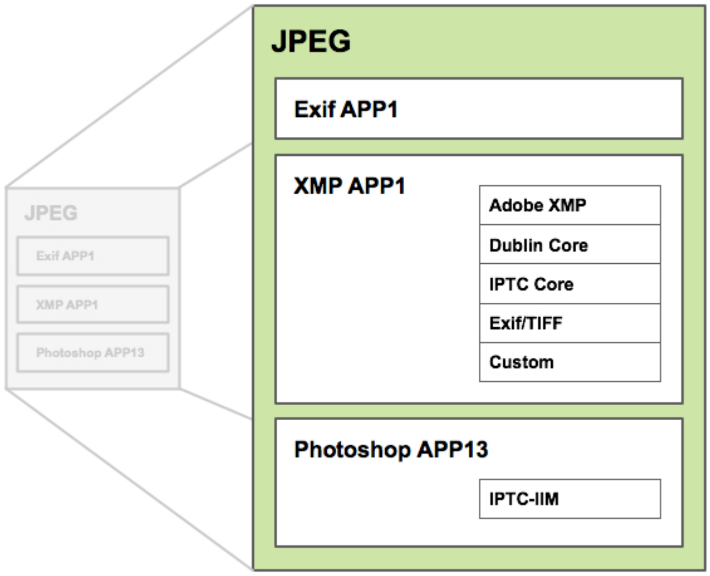
Digital images are stored in a variety of common file formats such as TIFF, JPEG and PSD as well as proprietary formats such as RAW. Each file format has unique rules regarding how metadata formats must be stored within the file.

Within image file formats, metadata can be stored within a variety of common metadata container formats such as Exif/TIFF IFDs, Adobe XMP, Photoshop Image Resources (PSIR) and IPTC-IIM. Each metadata container format has unique rules regarding how metadata properties must be stored, ordered and encoded within the container.

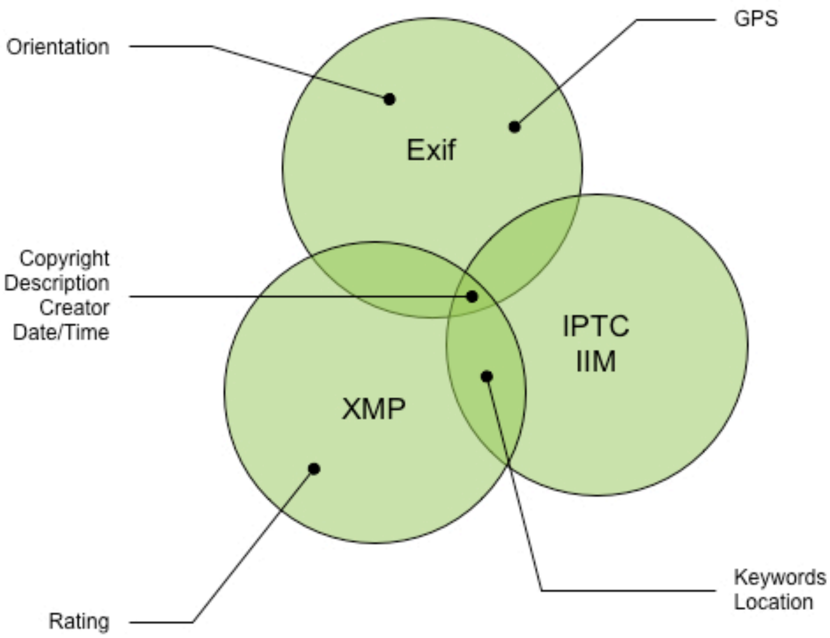


Within metadata container formats, metadata can be stored within a variety of semantic groupings. Examples of these groupings are Exif's GPS, XMP's Dublin Core, and IPTC-IIM's Application Record.

Some metadata semantic groupings, such as IPTC's, are targeted for specific user workflows. Some metadata semantic groupings, such as Exif's, can be stored within multiple metadata containers.



Within metadata semantic groupings, there can be dozens of individual metadata properties. Each metadata property can require data of specific types such as strings, numbers or arrays. Some metadata properties are conventionally read-only while other can be user modified. Some metadata properties are typically objective while others are subjective. Some useful properties, such as user ratings, have no commonly used standard storage container while others, such as copyright string, can be stored within many containers with similar but subtly distinct semantics.



The above structural complexities have traditionally caused further complications which challenge the effective use of metadata in workflows:

- Different applications and devices have chosen different policies in cases where standard metadata specifications were weakly or ambiguously defined.
- Different applications and devices have chosen different policies in cases where metadata can be stored in more than one standard location.
- An application or device often stores proprietary metadata, such as maker notes, within a metadata container. This practice is fragile because such private data can easily be lost when a different application modifies a file.
- Some applications and devices usurp a general purpose metadata property to address a specific need. This can cause compatibility problems for applications that correctly use the property in accordance with the generally accepted specification,
- Some applications avoid the complexities of storing metadata within image files altogether and opt instead to store it in a separate file or database. This practice can easily result in the loss of metadata when a file is used across several applications.

All of these problems have lead to significant frustration to users who want consistent metadata interoperability across digital imaging products and services. Manufacturers of digital imaging hardware, software and services spend substantial development resources dealing with these problems. Until practical guidance to resolve these complexities exists, these problems will continue to cost both users and industry time and resources.

1.1 Introducing the Metadata Working Group

Based on a 2006 proposal by Microsoft, the Metadata Working Group (MWG) organization was created in 2007 by 5 founding members: Apple, Adobe, Canon, Microsoft and Nokia. Sony joined this initiative in 2008.

The goals are:

- Preservation and seamless interoperability of digital image metadata.
- Interoperability and availability to *all* applications, devices, and services.

The organization is based on a formal legal framework and royalty free intellectual property policy that allows member companies and other industry leaders to collaborate on a solution to the above problems. The efforts of the MWG are organized into initiatives. The first initiative (embodied in this document) addresses digital imaging metadata for typical consumers. Future initiatives might include metadata for professional photography, audio and video metadata, etc.

1.2 Scoping the work

Consumer sharing of still images has exploded with the maturing of Internet services for the storage, manipulation and sharing of pictures. However, the majority of standards related to still images are oriented toward the documentation of the creation of an image or towards professional (e.g. print media) usage and management of images. There are no “advocates” in the ecosystem for the consumer who simply wants to share with friends, manage her snapshots, or deal with photos in unique, personal ways. The intent of this document is to use the existing standards to address the key organizational metadata questions that most consumers have:

- **Who** is involved with this image (who took it, who owns it, who’s in it)?
- **What** is interesting about this image?
- **Where** is this image from?
- **When** was this image created or modified?

The goal of this document is to provide best practices specifically for these critical data, with the intent of solving interoperability issues in the consumer space.

When we look at the “4W’s” (who, what, where, when), it is clear that this data can range from highly precise (e.g. a GPS latitude/longitude) to extremely vague or context dependant (e.g. “In my backyard”). It is not the intent of this document to solve the difficult semantic issues around this problem, but rather to help ensure that semantically equal metadata is identified across standards, and where it exists, semantically well-defined properties are chosen as best practice containers for the data. The two key notions of “reconciliation” and “rationalization” for the consumer space, defines the scope of this initial work.

1.3 Digital imaging metadata initiative

The scope of this initial effort is targeting still-imaging metadata, with a focus on consumer workflows (i.e., at this time it does not specifically address professional scenarios). The effort has been highly focused on solving a specific set of problems, outlined below. Future releases of this initiative will both refine and expand the effort.

Specific issues addressed in this document

- Interoperability and preservation of metadata between processes (devices, applications, platforms and services), file formats and metadata standards.
- Overlapping fields between existing standards and schemas.

These issues have been addressed by the creation of:

- A usage and data model based on common consumer use cases.
- Actor definitions of the roles each device or application play when interacting with metadata.
- Best practices for how, when and where metadata should be changed in popular consumer still image file formats using existing industry metadata standards.
- Rationalization of common and important consumer metadata fields between existing standards.

Specific non-goals for this document

- To define new metadata structures, storage formats, or standards where ones exist today.

1.4 Relationship to standards organizations

There are a number of established standards, such as Exif and IPTC-IIM, widely used by the digital photography industry. This effort is not intended to replace existing industry standards, but rather to build on them by providing resources to *improve interoperability* and *metadata preservation* between them. This is based on significant understanding of the industry (customers, scenarios, technologies) and experience in building the products that capture, process, store, share and transmit digital photographs.

This document, “Guidelines For Handling Image Metadata”, is designed to help guide developers by providing best practices on how to create, read and modify metadata within digital images. It’s also designed to motivate the owners of metadata standards and formats to think about preservation, interoperability, and compatibility in general terms.

1.5 Definition of terms

Digest	A checksum value to help identify changes between the metadata formats
Dublin Core	The Dublin Core is a metadata element set. It includes all DCMI terms (that is, refinements, encoding schemes, and controlled vocabulary terms) intended to facilitate discovery of resources.
Exif	“Exchangeable image file format” – standard for image file formats introduced by Japan Electronics and Information Technology industries Association (JEITA)
IPTC	“International Press Telecommunications Council” – creator and maintainer of metadata standards
IPTC-IIM	“Information Interchange Model” – IPTC multimedia metadata standard
IPTC Core	“IPTC Core” – IPTC photo metadata standard based on XMP
IPTC Extension	“IPTC Extension” – IPTC photo metadata standard based on XMP
JPEG	The JPEG file format, widely used in image and photography workflows
MWG	“Metadata Working Group” – Industry consortium responsible for this document
PSD	The native Adobe Photoshop file format
RDF	The “Resource Description Framework (RDF)”, described by the W3C as a “framework for representing information in the Web”, has become a general model for representing metadata
TIFF	The “Tagged Image File Format (TIFF)” is a file format to store images as well as photography.
Unicode	Unicode is an industry standard to consistently represent characters and text within modern software and hardware systems
UTF-8	UTF-8 is a byte-oriented encoding form of Unicode
XMP	“Extensible Metadata Platform” – multimedia metadata standard introduced by Adobe

2. USAGE AND DATA MODEL

For a better understanding of the proposed guidance in this document this chapter introduces the notion of different *actors* that play specific *roles* in metadata processing. These definitions will be used throughout the document to discuss the rules on how to handle metadata across the different formats.

2.1 Actor definition

In the actor model, the flow of an image file is represented as a series of phases between multiple applications (actors). It starts from the **Creator** actor, going through **Changer** actors and ending at the **Consumer** actor. In this model, all actors are essentially black boxes where processing actions, specific to the actor, are not known nor considered important from the model's point of view. However, the state of the metadata in the image file is communicated between each phase.

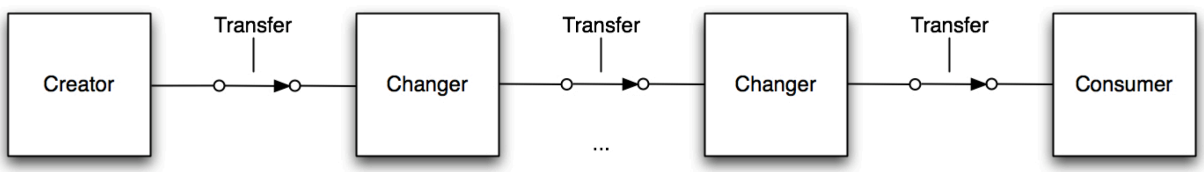


Figure 1 - Actor state diagram

An application is defined to be **compliant** if it reads and writes metadata in accordance to this document. There may also be non-compliant applications modifying metadata between two actors. It is not always possible to detect such a modification, so any compliant application must also accept non-compliant metadata.

This document presents the rules on how to handle a small set of selected metadata fields in a compliant manner. Roles defined in this document have two important functions: first they attempt to clarify the purpose of the selected fields, and secondly how to apply similar metadata handling to fields not covered here. An application can function in different roles at different times but every time it touches metadata, it does so only in one of these roles.

2.1.1 Creator

A **Creator** application creates the first instances of metadata into the (image) file. It is usually (though not necessarily) the same application that creates the image data, e.g. an image processing application, a digital camera or a cell phone. The typical aspect of a **Creator** application is that there is no old metadata to preserve. Alternately, an image editing application might behave as a creator even though it produces a new file from an existing file.

A **Creator** must fulfill these criteria:

- It MUST have full knowledge of all metadata it is creating.

- It MUST always create standard compliant metadata at least in one form as specified in this document.

In summary, the **Creator** understands all metadata it is writing and has exclusive access to the image content. There's no existing data the Creator needs to worry about. Most of the metadata is being determined while creating the content.

2.1.2 Changer

A **Changer** application first reads metadata from the image file and then writes new or modified metadata back to the same file.

The rules for an application in **Changer** role are:

- Deletion of metadata MUST only be done with specific intent.
- It SHOULD obey rules for **Consumer** applications when reading metadata.
- It SHOULD keep all forms of metadata it modifies in sync with each other.

The first rule is about preservation of metadata, which is a high priority. Descriptive metadata, information added by a user, MUST only be deleted by explicit user intent. Special attention SHOULD be paid to formally sensitive information such as a copyright. Non-descriptive metadata MAY be deleted by explicit user intent, SHOULD be modified or deleted if it is known to be inaccurate, otherwise it SHOULD be preserved. The second rule comes from the fact that, almost always, the **Changer** application is also a **Consumer** application so it must also observe all **Consumer** application rules. The third rule states, that whenever, the **Changer** application writes new metadata fields to the file, it must keep different forms existing in the file, e.g. Exif, IPTC-IIM and XMP, in sync. The first rule and third rule also apply to a changer that wishes to not write all forms of a metadata item. Writing one form and deleting other forms is a legitimate intentional deletion, done to not have unsynchronized forms in the file.

2.1.3 Consumer

A **Consumer** application only reads metadata from the image file. It may use metadata for display purposes, searching, content organization, etc. but it never modifies the metadata in the file itself.

General rules for **Consumer** application are:

- It MUST reconcile between different forms of metadata in the image.
- It MUST use metadata according to the semantics defined for each field.

The first rule says that a **Consumer** application must process metadata according to policies defined in this document and then only the reconciled data is used further before it is presented to the end-user. This may involve, for example resolving possibly conflicting information between Exif, IPTC-IIM and XMP versions of a metadata field existing in the image file. The second rule says that applications must understand the semantics of desired metadata fields and use them appropriately. For example, in order to reconcile different forms of metadata, the application must know the semantics of the metadata in question. In summary, the **Consumer** application treats image files as read-only so the state of the metadata remains unchanged.

Tools designed to display technical details about the format and content of the file's metadata, but not intended to primarily express metadata semantic meaning, are not required to be compliant **Consumer** applications.

3. METADATA MANAGEMENT

Metadata is an essential part of image and photography based workflows. Cameras capture device metadata while taking pictures. Operating systems and other software subsequently read metadata to build up catalogs and offer effective search capabilities. In addition, the user is able to enhance this workflow with its own metadata that will be stored either inside the file or within caching or database systems.

In the context of consumer image-based workflows, the existence of different metadata standards leads to interoperability issues when using various devices, operating systems and software tools. Although the majority of metadata properties are unique there are a number of properties that overlap across several metadata standards and thus lead to interoperability issues.

The goal of this section is to identify those overlapping properties and provide guidance on how to handle them correctly across the different metadata formats.

After a brief overview of existing metadata standards, this chapter introduces the most common metadata properties in the context of consumer workflows. To ensure best interoperability across software and hardware systems a general reconciliation mechanism will be discussed subsequently. Finally, the chapter will close with a detailed analysis of each focus area and discuss specific technical issues and obstacles.

3.1 Existing metadata standards

This section gives an overview of the existing metadata formats. As described in the introduction, this document will focus on photography workflows in the context of the consumer, so the choice of discussed metadata formats covers Exif, IPTC-IIM and XMP.

Exif - Exchangeable image file format

The Exif standard has been created by the Japan Electronics and Information Technology industries Association (JEITA[1]). In particular, the Exif image interchange format defines a set of TIFF tags that describe photographic images, and is widely used by digital cameras. Exif metadata can be found in TIFF, JPEG, and PSD files.

DCF - Design rule for Camera File system

As digital still cameras (DSC) have come to enjoy wide popularity, there is a growing need for direct exchange of images between cameras and other equipment, allowing pictures taken on one camera to be viewed on another, or to be output to a printer. The DCF specification is aimed at the creation of a user environment in which consumers can combine products more freely and exchange media readily. To this end it specifies rules for recording, reading and handling image files and other related files used on DSC or other equipment. Amongst others, DCF defines a subset of Exif where some properties are optional in Exif but required in DCF.

⇨ http://www.jeita.or.jp

[1] JEITA is the new name of the former Japan Electronic Industries Development Association (JEIDA)

IPTC-IIM – IPTC Information Interchange Model

IPTC, based in London, UK, is a consortium of the world's major news agencies, news publishers and news industry vendors. It develops and maintains technical standards for improved news exchange that are used by virtually every major news organization in the world.

In 1979, the first IPTC standard was text-only and defined to protect the interest of the telecommunications industry.

⇨ http://www.iptc.org

Later, in 1991, a new standard, the **“Information Interchange Model” (IIM)**, was created. It’s an envelope format for transmitting news text documents and photos and defining the so-called “IPTC headers” in many photo files, inserted by Adobe Photoshop and similar software.

⇨ http://www.iptc.org/IIM

After Adobe had introduced XMP in 2001, the IPTC Core standard has adopted XMP as the successor to the IIM-based “IPTC header” used to describe millions of professional digital images.

⇨ http://www.iptc.org/photometadata

XMP - Adobe's Extensible Metadata Platform

XMP is a labeling technology that allows you to embed metadata into the file itself. With XMP, desktop applications and back-end publishing systems gain a common method for capturing, sharing, and leveraging this valuable metadata - opening the door for more efficient job processing, workflow automation, and rights management, among many other possibilities. XMP standardizes the definition, creation and processing of extensible metadata.

XMP defines a metadata model that can be used with any defined set of metadata items. XMP also defines particular schemas for basic properties useful for recording the history of a resource as it passes through multiple processing steps, from being photographed, scanned, or authored as text, through photo editing steps (such as cropping or color adjustment), to assembly into a final image. XMP allows each software program or device along the way to add its own information to a digital resource, which can then be retained in the final digital file.

XMP is serialized in XML and stored using a subset of the W3C Resource Description Framework (RDF). Therefore, customers can easily define their own custom properties and namespaces to embed arbitrary information into the file.

⇨ http://www.adobe.com/products/xmp

3.2 Important metadata for the consumer

This section will introduce the most relevant metadata fields in the consumer workflow today. The selection will mainly serve the purpose of discussing the most important metadata fields, but due to the fact that information in these areas can be found in multiple metadata sources, it will also act as a model for other properties as defined in Exif, IPTC-IIM and XMP.

Keywords

Keywords are widely used across software applications today and are also called “tags” by some applications and services. Since so many existing applications allow for keyword display and editing it is now often misused. No longer strictly for keywords, applications overload the property with general-purpose information exchange such as for workflow or task management. In addition, recent cameras have the ability to assign tags automatically while shooting pictures. Keywords are usually user customizable although in the case of devices they are sometimes fixed.

Description

This area defines the textual description of a resource's content. Also known as “user comment”, “caption”, “abstract” or “description”. Today, this information is represented in different ways; sometimes integrated and displayed as one field – at other times revealed separately. This document combines the different sources into one overall representation, called “Description”.

Date/Time

There's a lot of confusion about date/time handling. In addition to a variety of date/time values stored within a file's metadata, there are also creation and modification values stored by the file system - both a computer's file system and that of a camera's media card.

In general, date/time metadata is being used to describe the following scenarios:

- Date/time *original* describes when a photo has been taken
- Date/time *digitized* describes when an image has been digitized
- Date/time *modified* describes when the user has modified a file

Date/time *original* and date/time *digitized* are usually added by the devices (cameras, scanners, etc.) but in other scenarios the user needs to define these values manually. This can happen for example if an old photograph is scanned in (digitized) and the user wishes to specify in the metadata the date the original photo was taken. The date/time *modified* value will be changed by software and operating systems on subsequent edits of the file.

This document focuses on the date/time *original* value, since that is generally the most important aspect for the consumer.

Orientation

One major pain point in image-based workflows is the correct handling of orientation. Today, various software tools do not follow a consistent rule in interpreting and changing the related metadata field in conjunction with the primary and/or thumbnail images - this leads to an incorrectly rotated display of the image. There are three scenarios of interest:

- Capturing orientation information on the devices
- Changing the orientation of an image by using an asset management tool
- Editing the image and rotating the pixels

These scenarios will be discussed in detail later and this document will provide clear guidance about how to handle orientation information.

Rating

The rating property allows the user to assign a fixed value (often displayed as “stars”) to an image. Usually, 1-5 star ratings are used. In addition, some tools support negative rating values (such as -1) that allows for marking “rejects” without deleting files in production.

Rating also can have fractional values. For example, online communities often deal with average values of rating coming from multiple users, which inevitably leads to fractional values.

Copyright

While copyright information has principally been the realm of the professional photographer, with the advent of easy online photo sharing sites, copyright is increasingly becoming important to the consumer as well. In the context of the consumer, this document focuses on two aspects:

- Copyright information
- URL to more information about the copyright

To avoid storing links as part of the copyright notice description, the optional copyright URL should be used to reference related information.

Creator

The creator, also known as “author”, defines one or more creators of an image. Some cameras allow embedding creator information on image creation.

Location

The location of an image is one of the key pieces of information that consumers want to capture. Until recently location capture was often accomplished with post-creation keyword annotation. With the advent of embedded GPS, accurate location information can now be automatically inserted into image files at creation time. Exif, IPTC-IIM, IPTC Core, IPTC Extensions and XMP all specify metadata properties that capture, with varying degrees of accuracy, either the location of the camera or the location of the image subject.

When storing location based information it's important to understand the difference between the two main concepts:

- "*Location Created*": This information describes the location where the image was created, the location of the camera during shot creation. The typical case is when a GPS receiver injects the current location into an image at shooting time (camera location).
- *"Location Shown*": This information describes the location of the main subject being shown in an image. For example, a picture of Mount Fuji would be tagged with the coordinates of where the mountain is located (subject location), although the picture may have been taken from downtown Tokyo.

In the latest specification (IPTC Core 1.1 / IPTC Extension 1.0), the IPTC now differentiates between camera location and subject location defining both "Location created" and "Location shown" as a set of hierarchical properties (World Region, Country Name, Province or State, City and optionally Sublocation). This resolves any issues around the semantics of location, and is clearly the way forward. E.g. Exif specification does not clearly differentiate between camera location and subject location in terms of GPS. In lieu of clear differentiation this document proposes policies in section 3.4.8 and 3.4.9 under "Specific core vocabulary and data issues".

In the **Consumer** context, the camera location and subject location have often been treated as the same thing. In the case where a semantic differentiation is made, it is very important to maintain these separate semantics.

Unlike keywords, which are unbounded, it is recommended that all location properties are entered to form a valid hierarchy and avoid ambiguity (e.g. simply filling the City property as "Springfield", or State/Province as "Victoria" will represent multiple locations).

3.3 Metadata formats within image files

There are three metadata formats widely used in the industry:

- Exif
- IPTC-IIM
- XMP

Within this section, guidance for reconciling and writing metadata across these metadata formats is discussed.

3.3.1 Handling a single metadata format

In the simplest scenario a given metadata property is only defined in a single metadata format. This is for example true for the rating property - this value should always be read and written into the corresponding XMP (xmp:Rating) field. No further reconciliation is necessary. Also, there are a variety of properties defined in Exif (device properties) or in IPTC-IIM (workflow properties) that are unique to the container and won't be reconciled amongst the other formats.

3.3.2 Handling multiple metadata formats

Dealing with more than one metadata format makes it challenging to determine the correct behavior on how to handle the particular property values. The main difficulty is the evolution of metadata representations and standards where older applications are not aware of newer practices. This can happen within a standard, such as the introduction of Unicode storage for IPTC-IIM. It can also happen across standards, such as with the introduction of XMP. Inconsistent implementations across software tools, encoding requirements, as well as size limitations on metadata properties cause additional challenges.

The properties described earlier have been identified as the most relevant in the consumer workflows today. However, they also serve another purpose in this document. Nearly all of them are defined in more than one metadata format, so they are good candidates to help understand the reconciliation issues between the various formats. In other words, if the problems for these properties are well understood, all other metadata properties can be handled accordingly.

Here is a simplified view of metadata for which guidance is being provided:

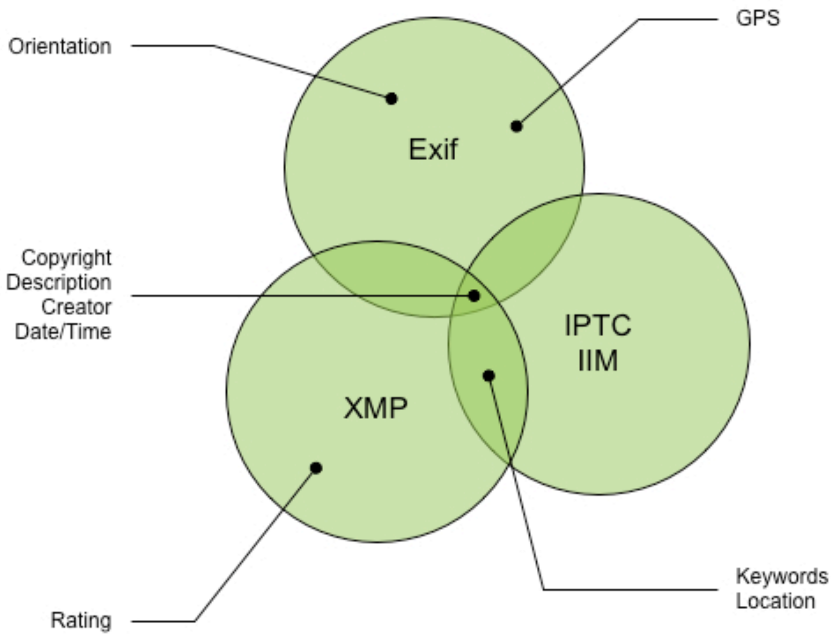


Figure 2 - Metadata defined in more than one format

It's noticeable that there are only a few properties defined in more than one metadata format. Actually only four are available in Exif, IPTC-IIM **and** XMP (Copyright, Description, Creator and Date/Time).

To ensure interoperability between existing and upcoming hardware and software solutions, the following sections will give you an overview on how to handle the different metadata properties in the context of the actor/role definitions.

But before going there, the next chapter will provide some more background information on the specific relationship between the metadata formats to better understand the overall picture.

3.3.2.1 Exif and IPTC-IIM in the context of XMP

The following diagram presents a different perspective on metadata usage:

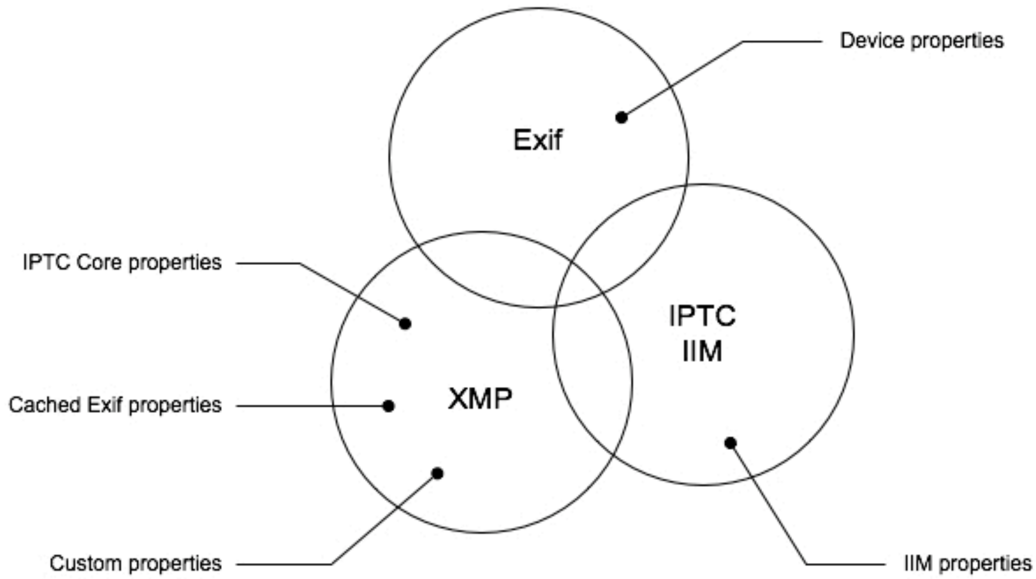


Figure 3 - Usage scenarios of Exif, IIM and XMP

Beside the fact that some native Exif and IPTC-IIM properties are mapped to corresponding XMP properties, some popular applications that pre-date this guidance also replicate a large number of other Exif properties into the XMP. To better understand the different use cases the following two chapters will put these properties into the context of this document.

Exif within XMP

The most recent (as of mid-2008) XMP specification describes the usage of Exif/TIFF properties within XMP itself. Both Exif (http://ns.adobe.com/exif/1.0/) and TIFF (http://ns.adobe.com/tiff/1.0/) namespaces have been defined so corresponding Exif properties can be stored. This is particularly useful if Exif properties need to be stored but the file format does not support native Exif (e.g. PNG).

In the case of file formats that **do** support Exif however, the current XMP specification describes mechanisms to reconcile data between the native Exif values and the mapped Exif properties in XMP (see “TIFF and Exif digests” under section “Reconciling metadata properties” in the XMP specification).

However, this document changes this earlier XMP guidance and recommends that Exif and Tiff device properties only be mapped into XMP in the case the file format does not support Exif natively. For more details, please see section “Handling Exif and XMP“ below.

IPTC within XMP

In contrast to the earlier IPTC-IIM specification, the most recent IPTC Core specification allows storing IPTC properties within XMP. Most of the properties are mapped to existing standard namespaces but for those where this was not possible a new namespace “http://iptc.org/std/Iptc4xmpCore/1.0/xmlns/” has been introduced. The IPTC group encourages people to move from IPTC-IIM to its newer IPTC Core / IPTC Extension standard.

With that said, this document focuses on the interoperability between existing applications and is mainly concerned about the reconciliation between the earlier IPTC-IIM standard and XMP, concentrating on the areas discussed in this document. There is no “reconciliation between XMP and IPTC Core” - everything is in XMP.

3.3.3 Metadata reconciliation guidance

The process of handling metadata values from the various metadata formats is basically divided into three different scenarios that will be discussed in the following chapters:

- Read/Write Exif and XMP metadata
- Read/Write IPTC-IIM and XMP metadata
- Read/Write Exif, IPTC-IIM and XMP metadata

3.3.3.1 Handling Exif and XMP

This chapter discusses reconciliation guidance between Exif and XMP:

Reading Exif and XMP

Only a few properties are actually mapped between Exif and XMP and therefore relevant for reconciliation. These are:

- Date/Time
- Description
- Copyright
- Creator

Since only a few properties are mapped between Exif and XMP, they are dealt with on a property-by-property basis. Unlike IPTC-IIM, as seen later, there is no advantage in using a checksum value to detect changes to the Exif. Especially for consumer use, there is generally no loss of fidelity when preferring Exif over XMP.

Here is a detailed look:

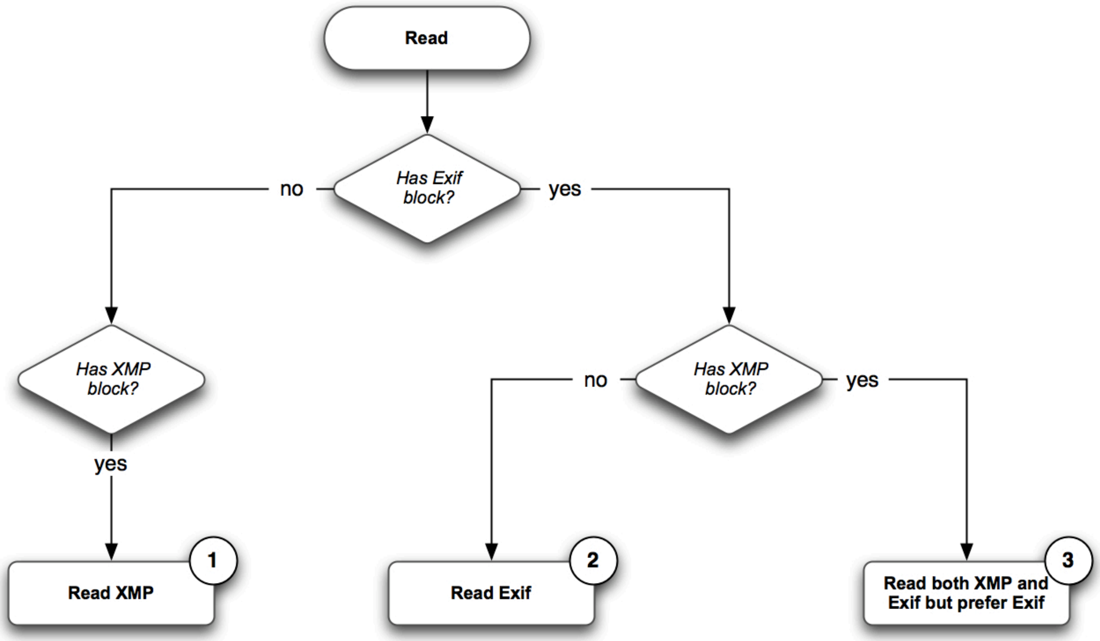


Figure 4 - Read guidance Exif/TIFF

If either Exif **or** XMP is available (1 & 2) reading metadata is straightforward.

Note: Today, there are two scenarios where Exif metadata is being mapped into XMP:

- Exif native properties mapped to respective XMP properties (e.g. Exif Copyright ⇔ XMP (dc:rights))
- Exif and TIFF device properties are being duplicated into specific “exif:” or “tiff:” namespaces (e.g. Exif ApertureValue ⇔ XMP (exif:ApertureValue))

In particular, for scenario (1) this means Exif and Tiff device properties SHOULD be read directly from the respective “exif:” and “tiff:” XMP namespaces. This is the case when the file does not natively support Exif.

The scenario “Read both XMP and Exif but prefer Exif” (3) is the most interesting one: As we'll see in more detail below, Exif can be preferred when reading because it does not have encoding or length limitations with respect to XMP. The policy for **Changers** ensures that newer applications write consistent values; preferring Exif when reading supports older applications.

It is however important to carefully read and follow the individual property mappings described in section “2.5 Specific Core Vocabulary and Data Issues” of this document. For example, the XMP (dc:description) value supports multiple languages, the corresponding Exif maps to a specific one of these.

The following diagram explains how to read Exif and XMP on a per property level in more detail:

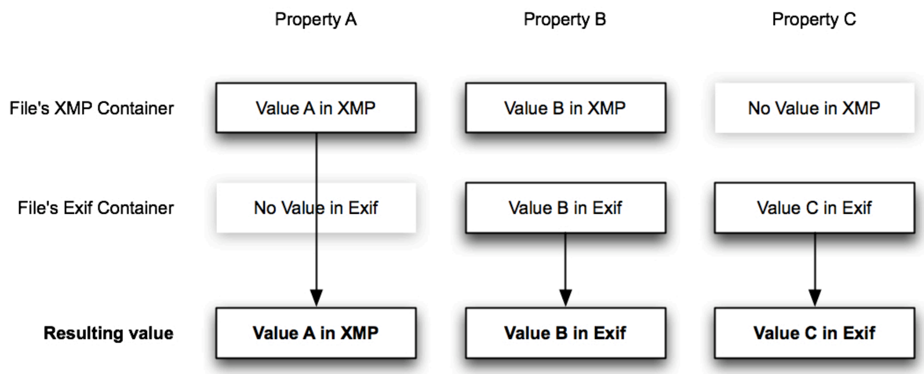


Figure 5 - Reconciling properties for Exif and XMP

For broader compatibility with non-compliant **Creators** and **Changers**, a **Consumer** SHOULD verify whether Exif text values are valid UTF-8. If not, a **Consumer** MAY assume the value is in a “local encoding” and convert it to UTF-8 as described under “Text encodings in read and write scenarios” below.

Writing Exif and XMP

In the context of the actor definitions the following rules describe the guidance on how to write XMP and/or Exif:

Creator

- XMP metadata MAY be created if a property can be written in both locations otherwise it MUST be created (which is true for file formats where Exif is not defined).
- If no XMP is written Exif metadata MUST be created.

Changer

- Exif and XMP metadata SHOULD be consumed according the reconciliation guidance described earlier (see “Reading Exif and XMP” above).
- When the file format supports both Exif and XMP, a **Changer** SHOULD update both forms of a value. If only one form is updated, an existing value in the other form MUST be removed.
- In the case the file format does support Exif natively, Exif and TIFF device properties (e.g. XResolution, YResolution, WhitePoint, etc.) SHOULD NOT be duplicated in the XMP exif: and tiff: namespaces.
- Exif metadata is formatted as a TIFF stream, even in JPEG files. TIFF streams have an explicit indication of being big endian or little endian. A **Changer** SHOULD preserve the existing endianness.

- Exif string values SHOULD be written as UTF-8. However, clients MAY write ASCII to allow broader interoperability[2].
- A checksum value for Exif/TIFF SHOULD NOT be written into the XMP.
- If no existing data is in the file the **Creator** guidance SHOULD be followed.

3.3.3.2 Handling IPTC-IIM and XMP

This chapter discusses reconciliation guidance between IPTC-IIM and XMP:

Reading IPTC-IIM and XMP

The use of IPTC-IIM is significant in professional workflows, and is also present in some consumer oriented tools. Although this document only directly addresses a few IPTC-IIM fields, there are several dozen in professional use. The IPTC-IIM values have length limitations and often character encoding issues that can make a conversion from XMP to IPTC-IIM be lossy.

For efficiency, and to avoid certain character encoding problems, a checksum (or digest) is used to detect overall changes to the IPTC-IIM values by non-compliant **Changers -** specifically those unaware of XMP. This checksum detects that something has changed in the IPTC-IIM block as a whole, but not specifically what changed. Further checks are then required to detect individual property changes.

The checksum value is an MD5 hash of the entire IPTC-IIM block, and is stored as a 16 byte binary value in Photoshop Image Resource (PSIR) 1061 (see "Writing IPTC-IIM and XMP" for more details). The checksum MUST be computed and stored when a **Creator** or **Changer** writes XMP and IPTC-IIM in sync. A **Consumer** MUST use the checksum as described below when reading XMP and IPTC-IIM.

[2] *It is understood that writing UTF-8 in Exif is formally in violation of the Exif specification, which requires 7-bit ASCII in most cases. Some devices (cameras and printers) will not be able to display non-ASCII characters.*

Now let's have a look at the **Consumer** guidance on how to read IPTC-IIM related metadata first:

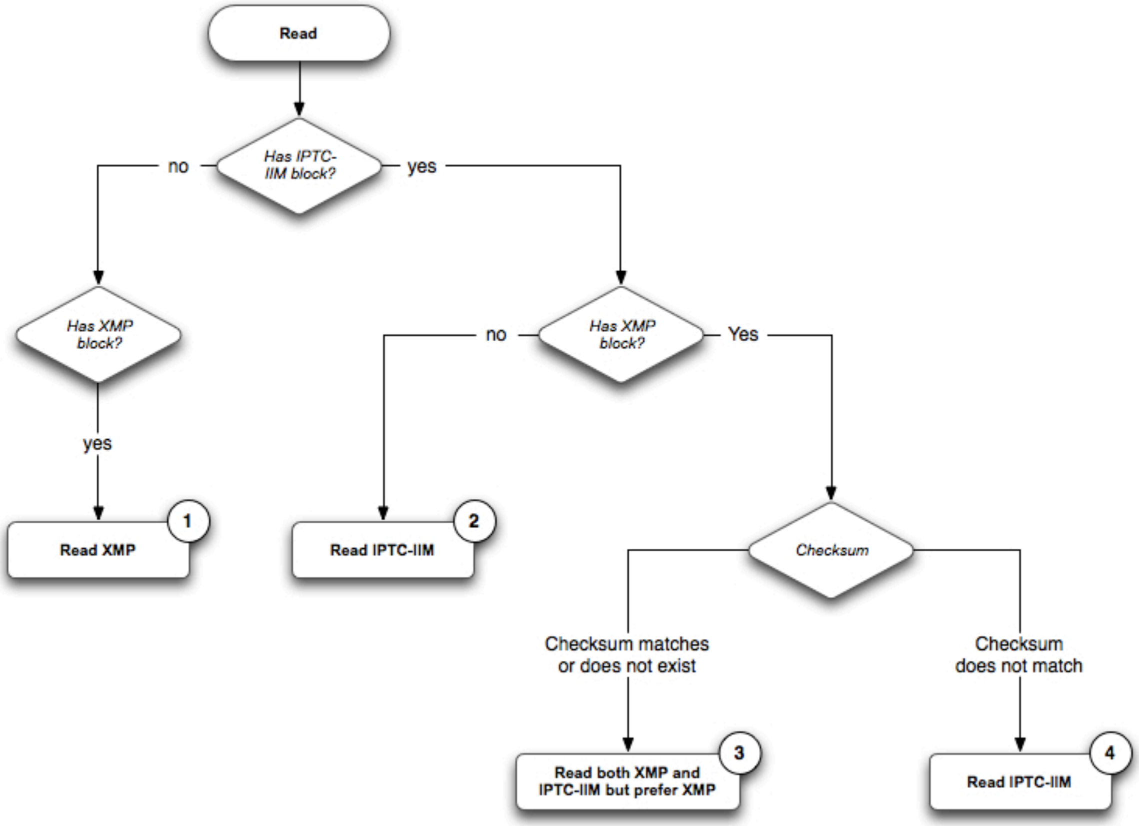


Figure 6 - Read guidance IPTC-IIM

In the case when either IPTC-IIM or XMP is available the read scenario is trivial (1 & 2).

However, scenarios (3) and (4) are more complex and require further explanation:

In scenario (3) either the checksum exists but matches the IPTC-IIM block **or** the checksum does not exist. In either case the following rules apply:

- Any existing XMP value is assumed to be more relevant and SHOULD be preferred.
- If an XMP value is missing then the IPTC-IIM value SHOULD be used.

The following diagram explains how to read IPTC-IIM and XMP on a per property level in this scenario:

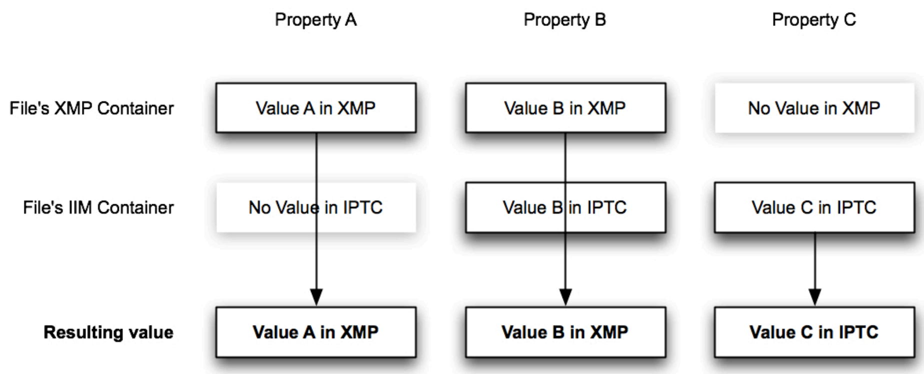


Figure 7 – Reconciling properties for IPTC-IIM and XMP

Finally, scenario (4) occurs when a non-compliant (XMP-unaware) **Changer** has exclusively modified the IPTC-IIM block. In this case a **Consumer** SHOULD check each property to decide if the IPTC-IIM or XMP value is used. This approach prevents loss of information in unchanged values due to truncation or character encoding. To do the check, the XMP value is used to create a predicted IPTC-IIM value, taking value truncation and character encoding into account. If the predicted and actual IPTC-IIM values match then the XMP value is used. Otherwise the IPTC-IIM value is used.

A **Consumer** MUST honor the encoding information provided by any IPTC-IIM Coded Character Set 1:90 DataSet that specifies UTF-8 - it SHOULD honor other 1:90 encodings. If this is not present, a **Consumer** MAY assume the value is in a “local encoding” and convert it to UTF-8 as described under “Text encodings in read and write scenarios”.

Writing IPTC-IIM and XMP

The following rules describe the guidance on how to write XMP and/or IPTC-IIM:

Creator

- SHOULD NOT create IPTC-IIM, unless it's required to be backward compatible with non-compliant **Consumers** that don't read XMP – otherwise SHOULD write XMP.
- If IPTC-IIM and XMP are both written, a **Creator** MUST create the checksum value as described earlier.

Changer

- XMP and IPTC-IIM SHOULD be consumed according the reconciliation guidance described above.
- If IPTC-IIM is already in the file, a **Changer** SHOULD write data back to the file in both XMP and IPTC-IIM – otherwise only XMP SHOULD be written.

- IPTC-IIM SHOULD be written using the Coded Character Set (1:90) as UTF-8 (see "Section 1.6 Coded Character Set" in the IIM specification).
- If the IPTC-IIM has not been written in UTF-8 before, a robust **Changer** SHOULD convert all properties to UTF-8 and write the corresponding identifier for UTF-8 to the 1:90 DataSet.
- If IPTC-IIM and XMP are both present, whether changed or not, a **Changer** MUST create or update the checksum value as described earlier.
- If no existing metadata is in the file the **Creator** guidance SHOULD be followed.

IPTC-IIM checksum

In the example of IPTC-IIM, the checksum MUST be calculated over the entire IIM block after values have been converted from XMP. The checksum itself MUST be stored in the Photoshop Image Resource (PSIR) 1061 resource as a 16-byte binary value representing the MD5 hash over the whole IIM block.

Example:

```
PSIR (1061) = 0ED63323337C50BF1E3BA76F6BB2122F
```

3.3.3.3 Handling Exif/TIFF, IPTC-IIM and XMP metadata

This chapter discusses reconciliation guidance between Exif, IPTC-IIM and XMP:

There are four properties that are defined in all metadata formats being discussed. Because the reconciliation guidance is specific for each property, please see section "Specific core vocabulary and data issues" later in this document for more details. If there's a conflict between Exif and IPTC-IIM, a **Consumer** SHOULD prefer Exif in the case the IPTC-IIM checksum matches or does not exist and SHOULD prefer IPTC-IIM in the case the checksum does not match. A string property that is comprised of only spaces or only nul characters MUST be treated as non-existent.

Upon writing Exif metadata, a **Changer** MUST update all formats that were originally present in the file. If not all of the formats were originally present, a **Changer** MAY choose to write the complete set.

3.3.3.4 More complex reconciliation in popular image formats

Finding and interpreting the metadata embedded in JPEG, TIFF, and PSD files is complicated by the fact that all three file formats contain the same kinds of metadata (XMP, Exif/TIFF, and IPTC-IIM), but store it slightly differently.

For example, all of the kinds of metadata can be contained in Photoshop Image Resources (PSIRs), and all three file formats (JPEG, TIFF, and PSD) can contain PSIRs. However, the specific contents of the PSIRs are different when contained in different image file formats. Each type of metadata is stored inside the PSIR for some file formats, and separately for others.

However, the recursive embedding of metadata formats is more a theoretical possibility, so this document will simplify this process by identifying the three most relevant places to find Exif, IPTC-IIM and XMP (highlighted below).

Here are illustrations of the various image file formats:

JPEG file format

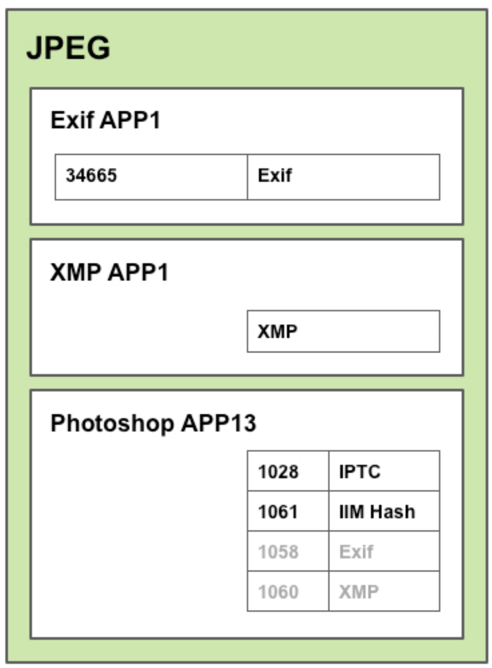


Figure 8 - JPEG file format

XMP SHOULD be read from the XMP APP1 section, IPTC-IIM SHOULD be read from the image resource block in Photoshop APP13 (1028) and finally Exif SHOULD be read from the Exif APP1 section. All other occurrences SHOULD be ignored. Please note: The APPn sections SHOULD be written according to the Exif specification. In particular, Exif APP1 MUST follow immediately after SOI. If this is not the case current camera models may not show Exif metadata correctly.

TIFF file format

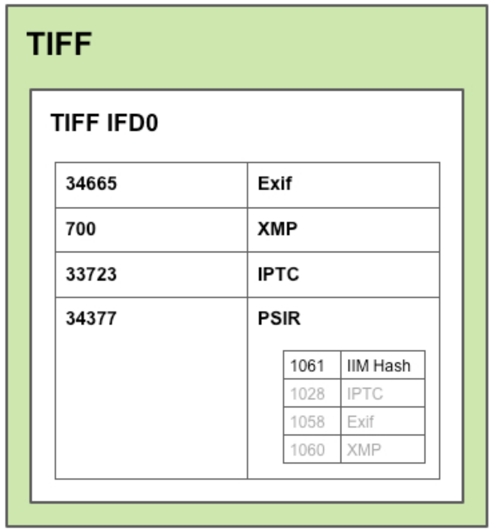


Figure 9 - TIFF file format

The *TIFF IFD0* contains the “Exif” (34665), “IPTC” (33723) as well as “XMP” (700) and SHOULD be used. The IPTC-IIM checksum is stored within the “PSIR” block (34377).

PSD file format

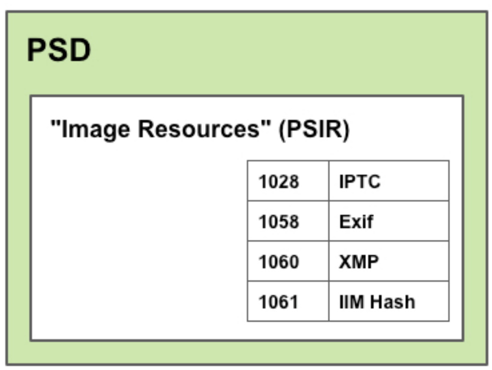


Figure 10 - PSD file format

The respective PSIRs - “Exif” (1058), “IPTC” (1028) and “XMP” (1060) SHOULD be accessed directly to read and write the various metadata formats.

Obviously, there are other file formats used by consumers including GIF, PNG, DNG, HD Photo, etc. These files will not be discussed in this document.

Text encodings in read and write scenarios

It is important to understand text encoding issues when reading and writing string metadata properties. The encoding defines the mapping between numerical byte values and user readable glyphs. It also defines the limits on what glyphs of which languages a byte sequence can represent. It is critical to know the encoding of a string property in order to correctly display the string to the user. If a string is displayed with the wrong encoding it will likely appear as a non-sensical string of glyphs.

Properties in the XMP metadata container format are always written using Unicode, generally UTF-8 encoding, which has the benefit that virtually any glyph of any language can be stored.

Non-XMP metadata property strings can be stored in a variety of encoding formats such as 7-bit ASCII, ISO Latin-1, JIS, or even UTF-8.

In some non-XMP metadata containers, the encoding is stored in the container along with the metadata. For example, the Exif UserComment tag has a prefix that indicates the encoding. Another important example is the IPTC-IIM metadata container that optionally supports the Coded Character Set 1:90 DataSet indicating the encoding of all the string properties in that container. This document requires that compliant consumers MUST respect any stored encoding indicators such as the above examples.

For other metadata string properties the encoding may be undefined by the container specification. Or the encoding may be de-facto undefined because in practice, a large number of files exist which are stored in a variety of encodings. In these situations a compliant reader SHOULD use a reasonable heuristic to infer the encoding used.

This document recommends that the following heuristic SHOULD be used to infer the encoding of string properties when the encoding is undefined:

- Scan the string to see if all bytes are in the range 0..127.
  - If so, assume the string is ASCII.
  - Otherwise, scan the string to see if it is consistent with valid UTF-8.
    - If so, assume the string is UTF-8.
    - Otherwise, assume a reasonable fallback encoding.

The choice of a reasonable fallback encoding is application and workflow dependent. It can be determined by querying the locale information of the host device or the user's preference.

It is also worth mentioning that a byte sequence that is consistent with a valid UTF-8 byte sequence is not 100% guaranteed to have been originally encoded in UTF-8. Nevertheless, such a test is highly reliable and it is highly beneficial to users to allow UTF-8 encoded strings to be read from and written to properties with undefined encoding conventions.

When a compliant **Creator** or **Changer** writes string properties where the encoding is undefined, it SHOULD consider the above heuristic. This means that the **Creator** or **Changer** should encode strings as UTF-8 or 7-bit ASCII. Another encoding MAY be used to be compatible with legacy workflows but doing so will produce strings that compliant readers have a small chance of misidentifying the string as UTF-8.

Time-zone handling

The handling of date/time values, and especially time zones, is conceptually easy but requires some care to avoid confusing users. The potential problems are typified by the differing representations of date/time values in Exif and XMP. (For our purposes here the Exif sub-seconds portions are ignored, but they are of course incorporated in software conversions.)

Exif date/time values such as DateTimeOriginal do not contain time zone information. The camera is presumably in an appropriate local time when a photograph is taken, but there is no indication in the Exif metadata of what that time zone was. The photograph's time zone MUST NOT be presumed to be the same as that of a computer later used to process the photograph.

The XMP specification formats date/time values according to the W3C note “http://www.w3.org/TR/NOTE-datetime”. In this note a time zone designator is required if any time information is present. A date- only value is allowed. The XMP specification has been recently revised to make the time zone designator be optional.

The representation of time zone as an offset from UTC can be ambiguous with regard to daylight savings time (DST). While date information can provide a strong hint, the use of DST is not universal and the date checking is complicated by changing rules for the start and end of DST in various locations. These issues are beyond the scope of this document; they may be addressed in a future revision.

The following general behaviors are recommended for time zone handling:

- A **Consumer** MUST NOT arbitrarily add a time zone. E.g. when importing Exif DateTimeOriginal to XMP (xmp:CreateDate), use a zone-less form for the corresponding XMP value.
- A **Changer** MUST NOT implicitly add a time zone when editing values. It is okay to be explicit about time zones if desired. Consider the typical case of correcting DateTimeOriginal values for an incorrectly set camera time. This must not be implicitly done as though the new time were in the computer's time zone.
- If the Exif contains the GPSDateStamp and GPSTimeStamp tags, software MAY use that information to infer a time zone. This should be done with care, e.g. verifying that the DateTimeOriginal plus inferred offset is within a few seconds of the GPS date and time..
- When time zone information is available, XMP values SHOULD be stored using the local+offset form, not the “Zulu” form (for example, use “2008-04-30T12:34:56-06:00” instead of “2008-04-30T18:34:56Z”). The local+offset form carries additional information, the Zulu value is easily determined when needed, e.g. for sorting in a UI.
- A user interface MAY display time zone information if available; however, related functionality MUST NOT convert a time to the computer's local time for display.
- According to the Exif specification, missing information SHOULD be filled up with spaces in the Exif values.

In summary, time-zone information MUST NOT be implicitly added and existing values should be preserved.

3.4 Specific core vocabulary and data issues

This section will provide a more in depth discussion of the most relevant properties in the consumer workflow and describe their current existence in the different metadata formats as well as detailed reconciliation aspects.

Note the following:

- For simplification, properties are named “Exif” independent whether they have been originally defined by the Exif or TIFF specification.
- The notation ns:array [“x-default”] means the “x-default” item in an XMP language alternative array whereas ns:array [*] represents the whole array. Please refer to the XMP Specification for more details around semantics and policies.

3.4.1 Keywords

Description	Words or phrases to classify an image
Representation	Information for the keyword property is available in IPTC Keywords (IIM 2:25, 0x0219) and XMP (dc:subject[*])
Type	*See respective specifications*
Value	*See respective specifications*
Guidance	See chapters “Handling IPTC-IIM and XMP“. Keyword lists SHOULD be completely replaced while reconciling.
Restrictions	Each IPTC-IIM keyword is limited to 64 bytes.
Notes	IPTC Keywords is mapped to XMP (dc:subject); IPTC Keywords can be repeated, each mapping to one of the elements in the XMP (dc:subject) array. Keyword properties usually do not retain the semantics of the keyword value itself. E.g. the information that “San Francisco” is a location will be lost. XMP/RDF provides the ability to add qualifiers for each keyword to define such a semantic. For future extensibility, these attributes SHOULD be preserved on any keyword manipulation. Hierarchical keywords are not covered. However it's well understood that this is an important use case even in the context of the consumer and will be added to future versions of this document. There are existing solutions available e.g. Adobe Bridge, Adobe Lightroom as well as Microsoft Expression Media and Windows Live Photo Gallery that have introduced hierarchical keyword workflows specific to their needs.

3.4.2 Description

Description	A textual description of the content shown in the image
Representation	Information for the description property is available in the following properties: Exif ImageDescription (270, 0x010E) IPTC Caption (IIM 2:120, 0x0278) XMP (dc:description[“x-default”])
Type	*See respective specifications*
Value	*See respective specifications*
Guidance	See chapter “Handling Exif/TIFF, IPTC-IIM and XMP metadata“
Restrictions	Length limitation in IPTC-IIM is 2000 bytes
Notes	Exif ImageDescription, IPTC Caption, and XMP (dc:description) are mapped together. In XMP, the description can be represented in multiple languages. In this document only the “x-default” value will be discussed and used. Clients MAY support the full range of localized values.

3.4.3 Date / Time

Description	Handling for original date/time
Representation	Information for Date/Time (Original) is available in the following properties: **Original Date/Time** - *Creation date of the intellectual content being shown* Exif DateTimeOriginal (36867, 0x9003) and SubSecTimeOriginal (37521, 0x9291) IPTC DateCreated (IIM 2:55, 0x0237) and TimeCreated (IIM 2:60, 0x023C) XMP (photoshop:DateCreated) **Digitized Date/Time** - *Creation date of the digital representation* Exif DateTimeDigitized (36868, 0x9004) and SubSecTimeDigitized (37522, 0x9292) IPTC DigitalCreationDate (IIM 2:62, 0x023E) and DigitalCreationTime (IIM 2:63, 0x023F) XMP (xmp:CreateDate) **Modification Date/Time** - *Modification date of the digital image file* Exif DateTime (306, 0x132) and SubSecTime (37520, 0x9290) XMP (xmp:ModifyDate)
Type	*See respective specifications*
Value	*See respective specifications*
Guidance	See chapter “Handling Exif/TIFF, IPTC-IIM and XMP metadata“
Restrictions	Exif DateTime does not contain time-zone information
Notes	Changes to XMP (xmp:CreateDate), for example to fix an incorrect camera setting, SHOULD be exported to Exif. If both XMP (xmp:CreateDate) and Exif DateTimeOriginal are missing, but Exif DateTimeDigitized (36868, 0x9004) exists, Exif DateTimeDigitized SHOULD be used to create an initial XMP (xmp:CreateDate). This is also true in the case only IPTC DateCreated is available. Exif DateTime is mapped to XMP (xmp:ModifyDate). Any change to the file SHOULD cause both to be updated. The above guidance implies that Exif sub-second and IPTC-IIM time properties are being handled according to the corresponding main properties. DCF specification requires DateTimeOriginal and DateTimeDigitized; the Exif specification recommends DateTime.

3.4.4 Orientation

Description	Orientation of an image and its thumbnail
Representation	The Orientation is represented in Exif Orientation (274, 0x0112)
Type	*See respective specifications*
Value	*See respective specifications*
Guidance	An image **Creator** MUST include an orientation tag in the Primary Image if the image raster data is intended to be displayed in any orientation other than the Normal (value 1) case where the 0th row represents the visual top of the image, and the 0th column represents the visual left-hand side. An image **Creator** MAY include an optional Thumbnail Image in the file. In this case, the **Creator** SHOULD write the Thumbnail Image in the same orientation as the Primary Image. If the Thumbnail Image is not written with the same orientation, then the creator MUST include an appropriate orientation tag value in the thumbnail IFD. A **Consumer** MAY choose to respect the orientation metadata included in a file when presenting an image or its thumbnail to the user. If a **Consumer** chooses to respect orientation metadata, it SHOULD: ▪ Treat the Primary Image orientation a Normal (value 1) if the Orientation tag of the Primary Image is missing. ▪ Treat the Thumbnail Image orientation as the same as the Primary Image if the Orientation tag of the Thumbnail Image is missing. If a **Changer** alters the pixel content of the Primary Image it SHOULD update or remove the Thumbnail Image (if previously present) so that a compliant **Consumer** does not display an inappropriate thumbnail. If a **Changer** alters the orientation metadata of the Primary Image, the **Changer** should also update the orientation metadata (if previously present) of the Thumbnail Image (if previously present) so that a compliant **Consumer** does not display an inappropriate thumbnail.
Notes	The DCF specification states that a thumbnail MUST be stored in a fixed size of 160x120 pixel. The thumbnail MUST be cropped or padded with black to meet the 160x120 pixel size requirement regardless of the aspect ratio of the primary image. Please consult the DCF specification for further details and restriction on JPEG images and thumbnails.

3.4.5 Rating

Description	Rating value of an image
Representation	Rating values are only available in XMP (xmp:Rating)
Type	Floating point number
Value	[-1.0; 0.0 … 5.0]
Guidance	The XMP (xmp:Rating) field SHOULD be read/written directly from/to the XMP
Notes	The value -1.0 represents a “reject” rating. If a client is not capable of handling float values, it SHOULD round to the closest integer for display and MUST only change the value once the user has changed the rating in the UI. Also, clients MAY store integer numbers. If a value is out of the recommended scope it SHOULD be rounded to closest value. I.e. value > “5.0” SHOULD set to “5.0” as well as all values < “-1.0” SHOULD be set to “-1.0”.

3.4.6 Copyright

Description	Copyright notice and reference to related information
Representation	The CopyrightNotice information is available in the following properties: Exif Copyright (33432, 0x8298) IPTC CopyrightNotice (IIM 2:116, 0x0274) XMP (dc:rights). The CopyrightURL SHOULD be stored in XMP (xmpRights:WebStatement)
Type	*See respective specifications*
Value	*See respective specifications*
Guidance	See chapter “Handling Exif/TIFF, IPTC-IIM and XMP metadata“
Restrictions	Exif Copyright can contain 2 strings - creator and editor rights - separated by a nul (0x00) character; length limitation in IPTC-IIM is 128 bytes
Notes	Exif Copyright, IPTC CopyrightNotice, and XMP (dc:rights) are mapped together. The Exif Copyright information (creator and editor rights) MAY be concatenated by a linefeed character (0x0A) when stored in other formats.

3.4.7 Creator

Description	Creator or author of the asset
Representation	The creator is available in Exif Artist (315, 0x013B), IPTC By-line (IIM 2:80, 0x0250), and XMP (dc:creator)
Type	*See respective specifications*
Value	*See respective specifications*
Guidance	See chapter “Handling Exif/TIFF, IPTC-IIM and XMP metadata“
Restrictions	Length limitation in IPTC-IIM By-Line is a repeatable of 32 bytes each
Notes	Exif Artist, IPTC By-line and XMP (dc:creator) are mapped together. Individual names are separate items in the XMP (dc:creator) array as well as separate (repeated) IIM By-line tags. A client MAY also follow the Exif specification in using the recommended example (“Camera owner, John Smith; Photographer, Michael Brown; Image creator, Ken James”).

3.4.8 Location (Created)

Description	Location information about where the image has been created
Representation	GPS data: Exif GPS (34853:[1-6], 0x8825:[1-6])
Type	*See respective specifications*
Value	*See respective specifications*
Guidance	The Exif GPS tags can be read/written directly without any reconciliation being required
Notes	The Exif specification does not clearly specify if the GPS tag 1 through 6 are to represent *location shown* or *created*. In lieu of clear differentiation, this document proposes that this group of properties SHOULD be treated as *location created*. The IPTC Extension 1.0 specification introduces a new mechanism that clearly defines the difference between where an image has been taken (*location created*) and where the content being shown on the image is located (*location shown*). The properties are represented as a hierarchy and in the case of *location created* defined as follows: *Location created* IPTC World Region *Ext* (Iptc4xmpExt:LocationCreated:WorldRegion) IPTC Country *Ext* (Iptc4xmpExt:LocationCreated:Country) IPTC Province/State *Ext* (Iptc4xmpExt:LocationCreated:ProvinceState) IPTC City *Ext* (Iptc4xmpExt:LocationCreated:City) IPTC Sublocation *Ext* (Iptc4xmpExt:LocationCreated:Sublocation) Basically, this makes it possible to map Exif GPS data to these new properties. However, any transformation guidance is beyond the scope of the initial version of this document.

3.4.9 Location (Shown)

Description	Location information about the content being shown in the image
Representation	Structured textual location metadata: The following properties are being defined in the **XMP** specification. They are also being used by the IPTC Core specification (see below): XMP (photoshop:Country) XMP (photoshop:State) XMP (photoshop:City) The following properties are defined in **IPTC-IIM**. They do not differentiate between *location created* and *location shown* but are intended to contain the location where the image has been created. Moreover, no formal hierarchy is specified between them: IPTC Country (IIM 2:101; 0x0265) IPTC Province/State (IIM 2:95; 0x025F) IPTC City (IIM 2:90; 0x025A) IPTC Sublocation (IIM 2:92; 0x025C) The following properties are defined in **IPTC Core 1.0**. They are specified as a hierarchy and represent the location shown in the image: IPTC Country (photoshop:Country) IPTC Province/State (photoshop:State) IPTC City (photoshop:City) IPTC Location (Iptc4xmpCore:Location) The IPTC Core 1.0 properties are still defined in **IPTC Core 1.1** but explicitly being labeled as "legacy": IPTC Country *Legacy* (photoshop:Country) IPTC Province/State *Legacy* (photoshop:State) IPTC City *Legacy* (photoshop:City) IPTC Sublocation *Legacy* (Iptc4xmpCore:Location) Please note that the "Location" label has been renamed to "Sublocation" in this revision of the specification.
Type	*See respective specifications*
Value	*See respective specifications*
Guidance	The following table shows the general mapping between IPTC-IIM and XMP / IPTC Core 1.0/1.1: IPTC Country (IIM 2:101; 0x0265) ⬄ XMP (photoshop:Country) IPTC Province/State (IIM 2:95; 0x025F) ⬄ XMP (photoshop:State)

	IPTC City (IIM 2:90; 0x025A) ⇔ XMP (photoshop:City) IPTC Sublocation (IIM 2:92; 0x025C) ⇔ IPTC (iptc4xmpCore:Location) An image **Creator,** when adding textual location metadata, MUST do so using automated or validated entry methods. Location text directly entered by end-users MUST be treated as keywords. To avoid the introduction of location ambiguity when using textual location properties, a **Creator** or **Changer** that adds these properties SHOULD add properties such that the hierarchy of location is complete to the lowest level entry added. For detailed reconciliation guidance see “Handling IPTC-IIM and XMP”.
Restrictions	According to the IPTC-IIM specification, the Location, City and State text fields are limited to 32 bytes in length. Also, the Country text field is limited to 64 bytes in length. XMP and IPTC Core are not facing this limitation.
Notes	As described above, the IPTC Extension 1.0 specification introduces a differentiation between where an image has been taken (*location created*) and where the content being shown on the image is located (*location shown*). The properties are represented as a hierarchy and in the case of *location shown* defined as follows: *Location shown* IPTC World Region *Ext* (Iptc4xmpExt:LocationShown:WorldRegion) IPTC Country *Ext* (Iptc4xmpExt:LocationShown:Country) IPTC Province/State *Ext* (Iptc4xmpExt:LocationShown:ProvinceState) IPTC City *Ext* (Iptc4xmpExt:LocationShown:City) IPTC Sublocation *Ext* (Iptc4xmpExt:LocationShown:Sublocation) The Exif specification does not clearly specify if the GPS tag 19 through 26 are to represent *location shown* or *created*. In lieu of clear differentiation, this document proposes that this group of properties MAY be treated as *location shown*. Please note: These tags are a different from the set of tags that are normally being used to store the camera location in GPS as described in 3.4.8 above. So in summary, GPS tag 1 through 6 SHOULD be used to store *location created* information whereas GPS tag 19 through 26 MAY be used to represent *location shown* information. That said, any transformation or usage guidance of the properties mentioned in the notes section are beyond the scope of the initial version of this document.

APPENDIX A: REFERENCES

Metadata Standards

Exif	http://www.jeita.or.jp
IPTC	http://www.iptc.org
IPTC-IIM	http://www.iptc.org/IIM
IPTC Core for XMP	http://www.iptc.org/photometadata
IPTC Extension for XMP	http://www.iptc.org/photometadata
XMP	http://www.adobe.com/products/xmp

Metadata Specifications

Exif 2.21/DCF 2.0

Official Exif specifications are available in paper form and can be ordered on the JEITA website.

IPTC-IIM 4.1

http://www.iptc.org/std/IIM/4.1/specification/IIMV4.1.pdf

IPTC Core 1.0

http://www.iptc.org/std/Iptc4xmpCore/1.0/specification/Iptc4xmpCore_1.0-spec-XMPSchema_8.pdf

IPTC Core 1.1 & IPTC Extension 1.0

http://www.iptc.org/std/photometadata/2008/specification/IPTC-PhotoMetadata-2008_1.pdf

XMP

http://www.adobe.com/devnet/xmp/pdfs/xmp_specification.pdf

File Format Specifications

JPEG

http://www.jpeg.org/jpeg/

TIFF

http://partners.adobe.com/public/developer/en/tiff/TIFF6.pdf

PSD/PSIRs

http://www.adobe.com/go/psir

Miscellaneous

RFC2119

http://www.ietf.org/rfc/rfc2119.txt

RDF

http://www.w3.org/TR/rdf-schema

Dublin Core

http://dublincore.org/documents/dces

RFC2119

http://www.ietf.org/rfc/rfc2119.txt

Date and Time (W3C)

http://www.w3.org/TR/NOTE-datetime

APPENDIX B: IMPLEMENTATION NOTES

This appendix provides non-normative notes to help implementers follow the letter and spirit of the formal guidelines. The words "may", "should", and "must" are used here in lower case, with their usual informal meanings.

Policy for Creators and Changers

The language of sections 2.1.1 and 2.1.2 has some subtleties that might not be noticed on a first reading. The basic intent is that Creators need to write compliant metadata that they know to be correct, and Changers need to strive to preserve information while ensuring that changes leave the metadata relevant and consistent.

The last sentence of the first paragraph of section 2.1.1 is specifically meant as a "loophole" for editing applications that wish to be strict about their output: "Alternately, an image editing application might behave as a creator even though it produces a new file from an existing file." This is saying that it is a legitimate design choice for an editing application to behave as a creator of its output, rather than a changer of its input. Users will learn that and not be surprised.

In section 2.1.2, "deletion" really refers to "total elimination", or deletion of all forms. As one of the examples cites, it is OK to remove some forms while keeping others. Doing that might have broader compatibility effects, e.g. making the metadata hidden from readers of only the deleted forms. But such readers are by definition not MWG- compliant, an MWG-compliant writer is not required to cater to non- compliant readers.

The phrases "done with specific intent" or "by explicit user intent" are intentionally vague. There can be many kinds of intent. For example, a user might ask that a saved file be as small as possible, or be redacted for publication. It is a legitimate design choice for this to involve deletion of metadata. Ideally additional safeguards should be placed around sensitive metadata such as a copyright, but that is the purview of the application not the MWG.

Forms of metadata written by Creators and Changers

There is intentionally no specific guidance about which forms of metadata should be written by Creators and Changers. This of course only applies to those items that can appear in more than one form. Compliant readers are intentionally required to look for all forms, and to reconcile among multiple forms. This gives writers the freedom to write whatever forms they find convenient, so long as those forms in the file are consistent.

Of course the broadest compatibility among MWG-compliant and non- compliant readers will be attained by writing all forms. But it is not up to the MWG to advocate the value of that over other considerations.

Local encoding of text in Exif and IPTC-IIM

Several sections mention an undefined, non-Unicode, "local" encoding for text. Some applications, especially those from Adobe, utilize the Windows and Macintosh notions of a "current default" character encoding for this. This encoding typically relates to a user's choice of UI language, and can be modified on any machine at any time.

This is not always code page 1252 for Windows and MacRoman on Macintosh, those values typically apply only to the United States and portions of Western Europe. For Windows, this interpretation of local encoding is CP_ACP. For Macintosh, this is based on smSystemScript.

Internal and external modification times

Developers can understandably get confused between the internal modification time of Exif DateTime or XMP xmp:ModifyDate, and the external file modification time maintained by the file system. There is no intent that these be identical.

The intent is that software set the internal modification time time to the current local time at some convenient point in writing a file. Hopefully not significantly before closing the file, so the internal and external times will often be close. The mention of "current local time" is intentional, it is reasonable and appropriate for xmp:ModifyDate to contain the local time zone offset.

Note that the external file system time can be significantly different, for example if the file is on an external server in another time zone. This difference, and the possible "reset" of the external modification time if a file is copied, are some of the rationale for having an internal modification time.

No specific guidance is given for the choice of displaying, sorting, etc., based on the internal or external modification time. That should be appropriate to the context, aiming to give users the most value and least confusion. When looking at just images, it seems reasonable to use the internal image modification time. When looking at files in general, it seems reasonable to use the file system time.

APPENDIX C: SUMMARY OF CHANGES

Release notes version v1.0.1

2.1.2 Changer

Rephrase rule about metadata deletion, add more explanation.

3.3.3.1, under "Writing Exif and XMP"

Added a rule about TIFF stream endianness.

3.3.3.2, under "Reading IPTC-IIM and XMP"

Modified the rule about DataSet 1:90 to MUST honor UTF-8, SHOULD honor others. UTF-8 is known to be used and has obvious advantages. Other encodings are not known to be in significant use.

3.3.3.4, Under "Text encodings in read and write scenarios"

Revise the logic for determining text encoding, placing the ASCII versus UTF-8 choices in the correct order.

3.4.2 Description

Removed mention of Exif UserComment to simplify the Description mappings. The original motivation for including Exif UserComment came from the Exif specification's description of the ImageDescription tag: "A character string giving the title of the image. It may be a comment such as "1988 company picnic" or the like. Two-byte character codes cannot be used. When a 2-byte code is necessary, the Exif Private tag UserComment is to be used.". Early implementation efforts uncovered problems in existing files where a camera had set Exif UserComment (common in Nikon DSLRs), and the user later set an IPTC-IIM or XMP description. In these cases, favoring UserComment would cause loss of the "true" description. The use of UTF-8 for general Exif text removes the encoding motivation of the Exif specification's advice.

3.4.3 Date / Time

Added explicit mappings for each of the 3 kinds of date/time: original, digitized, and modified. This moved the XMP xmp:CreateDate association from the "original" set to the "digitized" set, and added XMP photoshop:DateCreated to the "original" set. This should cause no practical change for digital photographs: a UI should prefer Exif DateTimeOriginal, and Exif DateTimeDigitized is almost always the same anyway.

3.2 Important metadata for the consumer, under “Location”; 3.4.8 / 3.4.9 Location Created / Shown

Added clarification about camera location vs. subject location concept.