LACS Asset Classification Standard

Status	Final
Version	1.0.0
Last updated	2024-08-01
Maintainer	Lumi Foundation
Contact	std-lacs@lumif.org

Executive summary

The LACS Asset Classification Standard (LACS) is a straightforward, self-declaration system designed to bring transparency to the origin of digital assets. Recognizing the increasing role of AI in asset creation, LACS provides a simple way for creators to classify their assets as Human-Made (LACS-H), AI-Assisted (`LACS-A`), or AI-Generated (`LACS-G`). This empowers users to make informed decisions about the assets they incorporate into their projects, fostering a more transparent and ethical asset ecosystem. LACS is built for easy integration into existing asset store platforms and is maintained by the Lumi Foundation. It is intended to be used as searchable tags, even on platforms that do not officially support the standard.

In addition to providing transparency, LACS also assures customers that creators have considered the legal implications of using AI in their work.

Contents

1. Introduction

The LACS Asset Classification Standard (LACS) is a simple, self-declaration-based system for classifying the origin of digital assets. It aims to increase transparency in the asset creation process and help creators make informed decisions about the assets they use. LACS is an independent standard maintained by the Lumi Foundation and is designed for easy integration into existing asset store platforms.

By increasing transparency about the role of AI, LACS also helps build trust with customers, assuring them that creators have taken legal considerations of the use of AI generated content into account.

2. Scope

2.1 Included Assets

This standard applies to digital assets, including but not limited to:

- **3D Models:** Characters, creatures, vehicles, weapons, props, architectural elements, environmental elements (trees, rocks, etc.), and other 3D objects.
- **2D Textures and Images:** Material textures (diffuse, normal, specular, roughness, metalness, etc.), sprite sheets, UI elements, concept art, skyboxes, and other 2D images.
- **Audio:** Sound effects (SFX), music loops, ambient sounds, voice-overs, and other audio clips.
- **Animations:** Character animations, rigging data, object animations, morph targets, and other animation data.
- **Shaders:** HLSL, GLSL, and other shader code.
- **Blueprints/Visual Scripting Graphs:** Node-based logic and scripting setups (e.g., Unreal Engine Blueprints, Unity Bolt).
- **VFX:** Particle systems, simulations, and other visual effects.
- **Level Design Assets:** Prefabs, brushes, and other elements used for building game levels.

2.2 Exclusions

This standard does not apply to:

- Complete games or applications.
- Software tools or engines used for asset creation.

3. Classification Levels

LACS defines three classification levels for assets:

3.1 `LACS-H` (Human-Made)

The asset was created primarily through human effort, potentially with the assistance of software tools. The use of AI is permitted in this category only if it is limited to generating or manipulating elements that lack artistic value, are easily replaceable, and are defined as Inconsequential Algorithmic Elements (IAEs) (see Section 4)

3.2 `LACS-A` (AI-Assisted)

The asset was created through a collaborative process involving both human and AI, where the human has made the most significant contributions to the final product. The human provides creative direction and makes key artistic choices, while the AI assists in generating variations, refining details, or performing other tasks that may go beyond the scope of an IAE as defined in Section 4.

3.3 `LACS-G` (AI-Generated)

The asset was created primarily or entirely by AI, with minimal human input beyond initial prompting, parameter setting, or curation of AI-generated output.

4. Classification Guidelines

The following guidelines should be considered when classifying assets under the LACS standard:

- **Focus on the Creative Process and Intent:** The classification should reflect the creative process used to create the asset, specifically the degree and nature of human vs AI involvement, and the overall creative intent behind the asset.

- **Significant Contribution:** For the "AI-Assisted" category (`LACS-A`), both human and AI contributions should be significant and contribute meaningfully to the final asset's form, content, and aesthetic.

- Use of **Inconsequential Algorithmic Elements (IAEs)**, which is defined as an element (asset, technique, method, resource) that does not contribute significantly to the asset's unique creative expression, aesthetic, or meaning. The key to identifying an IAE is that its usage does not reflect any creative intent or artistic decision-making. These elements are typically simple, repetitive, or serve a purely functional purpose. They could be easily replicated by a human without specialized skill or replaced with similar elements without impacting the overall artistic intent. Even if AI is used to generate or manipulate these elements, their creation is considered inconsequential to the creative process.

  Examples of IAEs:

  - A simple grid pattern used as a base for a texture.
  - Basic geometric shapes used as placeholders.
  - Automatically generated UV coordinates that are not unique to the creative work.
  - Automatically generated vertex attributes
  - An AI upscaler used to increase texture resolution, as long as it is not used to create new features.
  - Color correction or adjustments made using an AI tool.
  - AI-generated trivial sphere mesh

  Examples that are NOT IAEs:

  - Using AI to generate a unique character design from a text prompt.
  - Using AI to create a complex texture with unique details and patterns.
  - Using AI to compose a piece of music from a prompt.

  NOTE: For clarity: Any use of AI that does not conform to the IAE description disqualifies the asset from `LACS-H` classification, but does not speak to as to whether LACS-A or LACS-G is more appropriate.

- **Transparency:** Creators should err on the side of transparency and choose the classification that most accurately reflects the asset's origin.

- **Good Faith:** Creators are expected to make classifications in good faith.

- **When in Doubt:** Classify the asset in the higher category (e.g., if unsure whether an asset is `LACS-H` or `LACS-A`, classify it as `LACS-A`).

- **Legal Compliance:** Creators are responsible for ensuring they have the necessary rights and permissions to use any AI tools or AI-generated content in their asset creation process. They must also ensure they have the right to distribute the resulting assets under the chosen license. It is the creator's responsibility to understand and comply with all applicable laws and regulations related to copyright, intellectual property, and the use of AI in creative work.

4.1. Complex assets and collections

Many digital assets, especially in fields like game development, are not monolithic creations but rather complex assemblies of various components. Similarly, asset stores often offer collections or bundles of assets. This section provides guidelines for classifying such assets under the LACS standard.

4.1.1. Complex Assets

A complex asset is defined as an asset composed of multiple distinct, separable elements that could potentially be created using different processes and thus have different LACS classifications. Examples include:

- A 3D character model consisting of a mesh, textures, and rigging data.
- A glTF file containing multiple meshes, materials, and animations.
- A VFX asset composed of multiple particle systems and simulations.

When classifying a complex asset, each significant component should be evaluated individually according to the LACS guidelines. If any non-IAE component of the asset is classified as `LACS-A` or `LACS-G`, the entire asset's classification is elevated to be at least `LACS-A`. If all components are classified as `LACS-H`, the overall classification remains `LACS-H`.

If a platform supports it, granular classification of individual components is encouraged, as exemplified in the glTF section of Appendix A.

4.1.2. Asset Collections

An asset collection is a group of multiple assets bundled and distributed together. Examples include:

- A Unity Asset Store package containing various models, textures, and sounds.
- A ZIP file containing multiple 3D models in different formats.

Asset collections, in their entirety, should not receive a single LACS classification. Instead, the presence and proportion of different LACS classifications within the collection should be clearly indicated in the collection's description. For example:

- "This package contains mostly `LACS-H` assets (85%), with some `LACS-A` assets (15%)."
- "This collection includes `LACS-H`, `LACS-A`, and `LACS-G` assets. See individual asset descriptions for details."

If possible, providing a detailed breakdown of the classification of each asset within the collection is highly recommended.

5. Self-Declaration

Asset creators are responsible for classifying their assets according to the LACS standard, including how to handle complex assets as described in section 4.1. The self-declaration should be included in the asset's metadata or description. The classification tag should be used in a way that it is easily searchable and accessible on any platform. This allows the LACS classification to be used even on platforms that do not officially support the standard. See Appendix A for examples of where to place the LACS tag.

The classification should use one of the following tags:

- `LACS-H` (for Human-Made)
- `LACS-A` (for AI-Assisted)
- `LACS-G` (for AI-Generated)

6. Asset Store Integration

Asset stores should provide a dedicated field in their asset submission forms for creators to indicate the LACS classification. This helps inform customers about the asset’s origin and, given the legal implications associated with AI-generated content, also helps protect them. The field can be a dropdown menu with the three classification options (`LACS-H, LACS-A, LACS-G`) or a text field that allows creators to enter the classification tag. Asset stores should also consider how they will support the classification of collections of assets that may contain assets with different classifications. The field should be clearly labeled and easily searchable, allowing users to filter and sort assets based on their LACS classification. See Appendix A for examples of where to place the LACS tag.

7. Relationship to other standards

LACS is an independent standard focused on classifying the creative process behind digital asset creation, specifically regarding the involvement of AI. It complements other standards that address different aspects of digital content.

For example, Content Credentials (C2PA) is a standard focused on verifying the authenticity and provenance of media to combat misinformation. While LACS classifies how an asset was made, Content Credentials provide a tamper-evident record of its edit history. LACS does not address whether an asset has been tampered with or manipulated, only the role of humans and AI in the initial creation process.

Creators and platforms may choose to use LACS alongside other relevant standards to provide a more comprehensive understanding of their digital assets. The maintainers of LACS are open to collaborating with other standards organizations to further develop and promote transparency in digital asset creation and usage.

LACS-labeled assets can support a creative work's OSST classification. Please refer to the Open Standard for Synthetic Transparency for further details.

8. Legal Considerations

The use of AI in asset creation raises complex legal questions, particularly regarding copyright and intellectual property. Creators who use AI tools or incorporate AI-generated content in their assets must ensure they understand and comply with all applicable laws and regulations.

It is the creator’s responsibility to determine whether they have the necessary rights and permissions to use any AI tools or AI-generated content in their asset creation process. This includes understanding the terms of service and licenses of the AI tools used, as well as any restrictions on the use or distribution of the generated output.

Creators must also ensure they have the right to distribute the resulting assets under the chosen license. This may involve obtaining appropriate licenses or permissions for any copyrighted material used in the creation process, including data used to train AI models.

The legal landscape surrounding AI-generated content is constantly evolving. Creators should stay informed about developments in this area and seek legal counsel if they have any questions or concerns.

By classifying an asset under the LACS standard, the creator affirms that they have, to the best of their knowledge, complied with all relevant legal requirements related to the use of AI in the asset’s creation. This provides an important assurance to customers about the asset’s legal standing.

9. Disclaimer

The Lumi Foundation is not responsible for verifying the accuracy of self-declared LACS classifications. This standard relies on the good faith of asset creators. Misuse of the standard may result in reputational damage within the community. The Lumi Foundation will maintain and update this standard and reserves the right to make changes to it, but will never change the meaning of a class once it is added to the standard.

This standard does not provide legal advice. Creators should consult with legal counsel if they have questions or concerns about the legal implications of using AI in asset creation.

Appendix A: Examples of LACS Tag Placement

This appendix provides examples of where to place the LACS tag within common file formats and asset store platforms.

A.1. File Formats

A.1.1. PNG, JPG, and other Image Files

For image files, the LACS tag can be placed in the “Description,” “Keywords,” or “Tags” field of the image metadata.

Example using ExifTool:

```
exiftool -Description="LACS-A" -Keywords="LACS-A, AI-Assisted" myimage.png
```

A.1.2. Model Files

For 3D model files, the LACS tag can be placed in the “Description,” “Comments,” or a custom user-defined property field, depending on the specific file format and software used.

A.1.2.1 glTF

glTF is a modern and increasingly popular format for 3D assets. It supports custom attributes within its extras system. You can add the LACS tag as a top-level extra or within an object’s extras.

glTF files can contain multiple components, such as meshes, materials, nodes, and animations. Each of these components can have its own LACS classification, allowing for granular labeling within a single file. This is achieved by placing the LACS tag within the `extras` field of the corresponding component. For instance, a glTF file could contain a human-made mesh and an AI-generated material, each tagged accordingly. This allows for detailed asset provenance information to be embedded directly within the glTF file.

Example to LACS-label whole glTF file:

```
{
  "asset": {
    "version": "2.0"
  },
  "extras": {
    "LACS": "LACS-A"
  }
}
```

Example to LACS-label single mesh:

```
{
  "meshes": [
    {
      "name": "MyMesh",
      "primitives": [...],
      "extras": {
        "LACS": "LACS-H"
      }
    }
  ]
}
```

A.1.2.2 FBX

FBX files can store custom properties within their User Properties section. You can add the LACS tag as a string property.

Note: The exact method for adding custom properties to an FBX file may vary depending on the software used.

A.1.2.3 OBJ

OBJ files do not natively support custom attributes. One workaround is to add the LACS tag as a comment at the beginning of the file, but this method is not standardized and might not be recognized by all software:

```
 # LACS: LACS-G
v 1.0 0.0 0.0
...
```

A.1.2.4 Blender

In Blender, you can add custom properties to objects, materials, and other data blocks.

Step-by-step instructions for adding the LACS tag to an object:

1. Select the object.
2. Go to the “Object Properties” panel.
3. Scroll down to the “Custom Properties” section.
4. Click “Add”.
5. Set the property name to “LACS”.
6. Set the property type to “String”.
7. Set the property value to the appropriate LACS tag (e.g., “LACS-A”).

Note: When exporting to formats like glTF, Blender will automatically include these custom properties in the extras section. For FBX, the custom properties should also be exported.

A.1.3. WAV, MP3, and other Audio Files

For audio files, the LACS tag can be placed in the “Comments,” “Description,” or “Genre” field of the audio metadata.

Example using FFmpeg:

```
ffmpeg -i myaudio.wav -metadata comment="LACS-G" -metadata genre="LACS-G" output.wav
```

A.1.4. Shader Files

For shader files, add a comment at the top of the file like this:

```
// LACS-H
```

A.1.5. Other File Formats

For other file formats not listed here, please consult the format’s documentation or the software’s help files to determine the best way to store custom metadata.

A.2. Asset Stores

A.2.1. Unity Asset Store and Unreal Engine Marketplace

These platforms have dedicated fields for tags and keywords. You should add the appropriate LACS tag (LACS-H, LACS-A, or LACS-G) to the tags or keywords field during the asset submission process.

A.2.2. Sketchfab

Sketchfab allows you to add tags to your models. Add the LACS tag in the “Tags” section when uploading or editing your model.

A.2.3. ArtStation

ArtStation has a “Tags” field when uploading artwork. Use this field to add the LACS tag.

A.2.4. CGTrader

CGTrader allows you to add both “Categories” and “Tags” to your 3D models. While there isn’t a dedicated field for LACS, it’s recommended to include the LACS tag within the “Tags” field.

A.2.5. TurboSquid

TurboSquid uses a “Keywords” field for product metadata. Include the relevant LACS tag (LACS-H, LACS-A, or LACS-G) in the keywords.

A.2.3. Other Marketplaces

For other marketplaces that do not have dedicated fields, creators should include the LACS tag as part of the asset’s description or in a way that allows it to be easily found in a search.