PLAYTEX AI FIELD MANUAL / 01

THE GAME DEVELOPER'S PBR TEXTURE HANDBOOK

A practical, engine-aware system for building believable, portable, and production-ready texture sets

COLOR

SHAPE

RESPONSE

Written by PLAYTEX AI Editorial Team

FLAGSHIP REFERENCE

The Game Developer's PBR Texture Handbook

A practical, engine-aware system for building believable, portable, and production-ready texture sets

WHAT THIS BOOK GIVES YOU

- A renderer-aware PBR mental model
- Map-by-map authoring and verification rules
- Calibration prompts for six surface families
- Engine import, naming, packing, and memory references
- A repeatable production QA and troubleshooting system

PLAYTEX AI Editorial Team

First edition - August 2026

How to read this authority guide

This handbook separates three kinds of guidance so you can tell what is fixed, what is target-specific, and what still requires judgment.

SPECIFICATION

A normative format rule, such as glTF channel layout or PNG alpha representation.

TARGET CONVENTION

An engine, pipeline, shader, or importer expectation that can differ by destination.

REVIEW WINDOW

An editorial starting range for visual review - never a universal material standard.

EDITORIAL STANDARD

Technical claims were checked against current first-party specifications and engine documentation available in August 2026. Always retest project presets after an engine or renderer upgrade.

Contents / chapters 01-09

01	**The PBR mental model**	8
02	**What each texture map represents**	12
03	**Base color versus diffuse color**	16
04	**Tangent-space normal maps and height maps**	20
05	**Roughness, smoothness, and reflection spread**	24
06	**Metallic classification and mixed surfaces**	28
07	**AO, emission, opacity, and displacement**	32
08	**Source-image preparation and delighting**	36
09	**Creating seamless materials**	40

Contents / chapters 10-18

10	**Building a coherent map stack**	44
11	**Material calibration by surface family**	48
12	**Reviewing maps under controlled lighting**	52
13	**Color space, bit depth, naming, and file formats**	56
14	**Channel packing and export conventions**	60
15	**Engine-import matrix**	64
16	**Performance and texture-budget fundamentals**	68
17	**Production QA checklist**	72

APPENDICES / PAGE 80

Material calibration cards / naming convention / engine import matrix / memory estimator / production QA / troubleshooting / glossary / PLAYTEX AI tools / sources

A four-pass material workflow

01

DESCRIBE

Name the visible substance, layers, physical scale, and target shader before touching values.

02

AUTHOR

Build clean canonical maps with shared masks, stable structure, and documented semantics.

03

TRANSLATE

Generate target-specific channel packs, inversions, formats, and importer settings.

04

VERIFY

Review under controlled light, then profile compression, mips, motion, and memory in target.

The evidence packet

A material is easier to approve, debug, and hand off when every revision carries the same small evidence packet.

01 **Canonical maps**

Unpacked source maps at approved precision and scale.

02 **Material manifest**

Semantic, color-space role, neutral value, channels, version, and provenance.

03 **Review captures**

Unlit, neutral, grazing, alternate environment, distance, and motion frames.

04 **Target preset**

Engine, pipeline, shader, normal convention, packing, format, and importer flags.

05 **Runtime record**

Resident memory, compression, mip behavior, streaming state, and exceptions.

CORE RULE

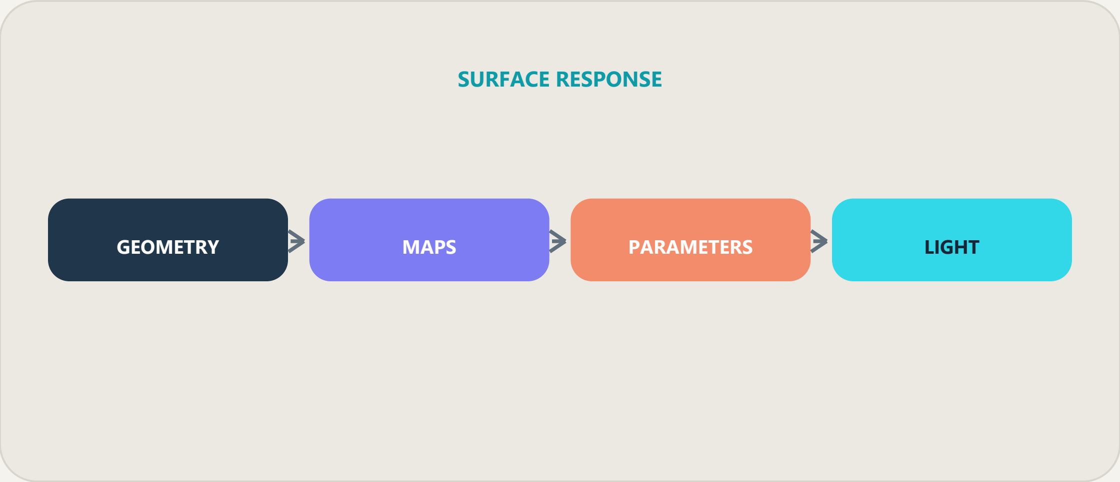
If a delivery cannot be regenerated from its canonical maps and recorded preset, it is not yet a production system.

CHAPTER 01

The PBR mental model

Describe the surface. Let the renderer describe the light.

Physically based rendering is a contract between material data, lighting, and a shading model. Your maps describe stable surface properties; the renderer evaluates how those properties respond to the current lights, camera, and environment.



KEEP THIS

A portable material does not paint the final picture into its textures. It supplies clean evidence about color, micro-surface shape, reflectance class, and coverage.

What to understand

1

Four layers, one result

Think in four layers: geometry defines the silhouette; maps define spatial variation; scalar parameters set global response; lighting reveals the result. If a problem appears only under one light, test the light before repainting the maps.

2

Energy has to go somewhere

A surface cannot return unlimited light. Dielectrics split energy between diffuse and specular response. Conductors suppress the diffuse lobe and tint reflected light with base color. This is why metalness is a classification, not a generic shine control.

3

Microfacets explain roughness

PBR shaders treat a surface as many tiny facets. Low roughness aligns reflections into a tight image; high roughness spreads them over a wider angle. Roughness changes reflection spread, not whether a surface reflects at all.

4

Consistency beats hero lighting

A sound material remains plausible in a neutral studio, an overcast exterior, and a high-contrast environment. Author against controlled references, then stress-test under different lighting instead of compensating for one flattering setup.

Production workflow

1. Name the real surface and separate coating from substrate.
2. Classify each region as dielectric, conductor, or a boundary between them.
3. Author base color without baked illumination.
4. Set roughness from the expected reflection spread.
5. Add normal and height detail only at the scale they can represent.
6. Review under a fixed neutral rig before creative lighting.

PLAYTEX AI ROUTE

AI HDRI Generator

Create alternate lighting environments for portability checks.

playtex.ai/hdri-sphere-generator

Verify before approval

PASS CHECKS

- [ ] Does the material still read when the environment rotates?
- [ ] Are highlights generated by the renderer rather than painted into base color?
- [ ] Does roughness control the width of reflections without muting all reflectance?
- [ ] Are metal regions physically distinct from dirt, paint, rust, or oxide above them?

COMMON FAILURE MODES

- Using metallic as a brightness slider.
- Darkening base color to hide a lighting problem.
- Judging the material under only one HDRI.

TWO-MINUTE CONTROL

Replace all maps with constants, then reintroduce one input at a time under a fixed light.

CHAPTER 02

What each texture map represents

Every map needs one declared job.

A texture map is not just an image. It is a field of values with a declared meaning, range, color-space role, channel layout, and destination input. Treating all maps as ordinary color images silently changes the data.

BASE COLOR	NORMAL
sRGB	LINEAR
ROUGHNESS	METALLIC
LINEAR	LINEAR

KEEP THIS

Before editing a pixel, write the map contract: semantic, value range, transfer function, neutral value, channel, resolution, normal convention, and target renderer.

What to understand

1

Color-bearing maps

Base color and most emissive textures encode visible color. They are usually stored with the sRGB transfer function and decoded to linear values for lighting. Alpha remains linear coverage data even when it shares a base-color file.

2

Data maps

Roughness, smoothness, metallic, ambient occlusion, height, masks, and tangent normals are numerical inputs. Sample them as linear or non-color data. Their thumbnails may look odd because human-friendly display is not their purpose.

3

Neutral values matter

Flat tangent normal is approximately RGB 128,128,255. White AO means unoccluded. Black metallic means dielectric. Roughness and smoothness are opposites, so their neutral artistic midpoint may match numerically but not semantically.

4

One source can feed many targets

Keep a canonical, unpacked source stack. Derive engine-specific packed textures and inverted channels during export. Editing packed delivery files by hand makes provenance unclear and invites stale channels.

Production workflow

1. Inventory every file and identify its semantic role.
2. Record color or non-color sampling for each input.
3. Mark the expected neutral value and valid range.
4. Check whether channels are independent or packed.
5. Record the target shader slot and normal Y convention.
6. Reject ambiguous names before the set enters production.

PLAYTEX AI ROUTE

PLAYTEX AI Material Preflight

Inspect dimensions, channels, and delivery risks as a set.

playtex.ai/tools/material-preflight

Verify before approval

PASS CHECKS

- [ ] Can a teammate identify every map without opening it?
- [ ] Are color-bearing maps separated from numeric data maps?
- [ ] Do packed channels have a documented order?
- [ ] Can the canonical sources regenerate every engine export?

COMMON FAILURE MODES

- Inferring semantics from a grayscale thumbnail.
- Saving an already packed map over a canonical source.
- Assuming alpha always means opacity.

TWO-MINUTE CONTROL

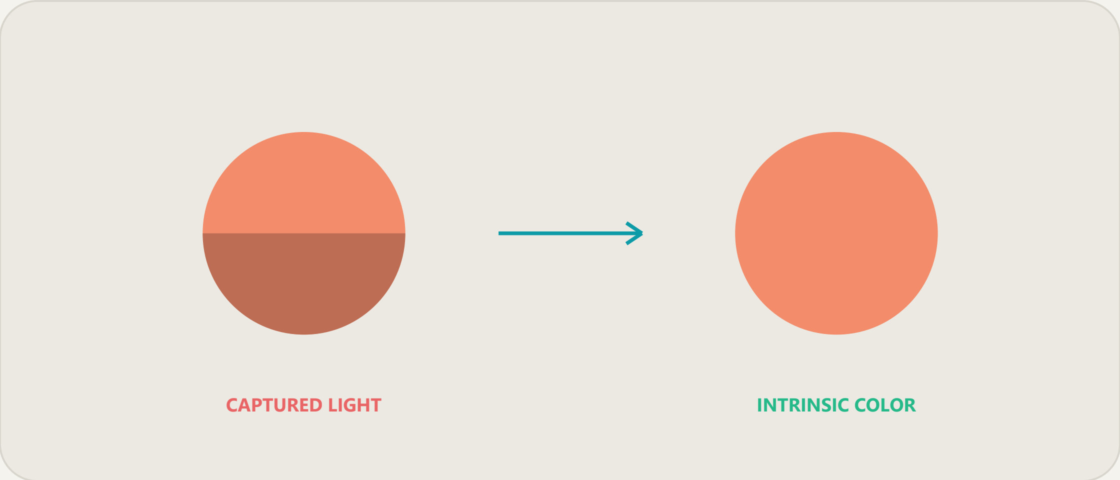
Set a data-map midtone to 0.5 and confirm the shader receives 0.5, not an sRGB-decoded value.

CHAPTER 03

Base color versus diffuse color

Color is not illumination.

Base color in a metallic-roughness workflow carries the surface color used by both dielectric diffuse response and conductor reflectance. Diffuse color in older or specular workflows is not always an interchangeable label.



KEEP THIS

For dielectrics, capture intrinsic color without direct highlights, cast shadows, ambient occlusion, or white-balance drift. For metals, base color represents tinted reflectance rather than a conventional diffuse layer.

What to understand

1

Remove lighting, preserve identity

Delighting should remove directional illumination while preserving real pigment, stains, wear, and broad material variation. A perfectly flat result is not the goal; illumination-free surface information is.

2

Metals use base color differently

At metallic 1, the shader uses base color to tint reflected light and largely removes diffuse response. A dark metal base color can be valid, but painted, rusted, dusty, or oxidized regions are usually dielectric layers and need their own classification.

3

Avoid impossible extremes

Near-black dielectrics can starve the renderer of believable diffuse response; clipped whites erase recoverable variation. Use measured references where available, compare histograms, and keep headroom for lighting.

4

Do not multiply AO by default

Ambient occlusion belongs in its own input when the target supports it. Permanently multiplying AO into base color makes cavities stay dark even when direct or bounced light should reach them and limits later tuning.

Production workflow

1. Neutralize white balance and exposure before material edits.
2. Separate broad illumination from real surface variation.
3. Remove specular highlights and cast shadows conservatively.
4. Check metal and dielectric regions independently.
5. Review the result unlit and then under a neutral PBR shader.
6. Save the cleaned source before tiling or channel packing.

PLAYTEX AI ROUTE

Material Delighter / True Albedo Cleaner

Remove captured lighting before map generation.

playtex.ai/tools/material-delighter

Verify before approval

PASS CHECKS

- [ ] Does the unlit map still look like the material rather than a rendered object?
- [ ] Do highlights move with the light instead of staying fixed?
- [ ] Are cavity shadows absent from base color unless they are real pigment?
- [ ] Are metallic regions interpreted as reflectance color, not diffuse paint?

COMMON FAILURE MODES

- Using a beauty photograph as finished albedo.
- Removing stains and pigment while removing shadows.
- Baking AO into every export automatically.

TWO-MINUTE CONTROL

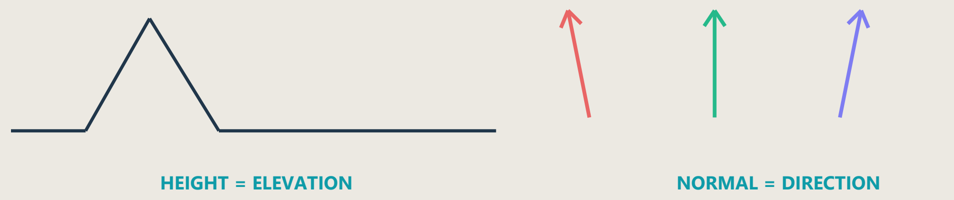
View base color unlit. Any directional highlight or cast shadow that remains is baked contamination.

CHAPTER 04

Tangent-space normal maps and height maps

Direction and elevation are related, not interchangeable.

A tangent-space normal map stores a direction per texel and changes lighting without changing the silhouette. A height map stores a scalar elevation field that can drive parallax, tessellation, displacement, or normal generation.



KEEP THIS

Normals are renderer-ready directional data. Height is source-like geometric intent. Preserve height when possible, then derive target-specific normals using the final scale and tangent basis.

What to understand

1

The tangent basis is part of the asset

A normal map is interpreted relative to mesh tangents, bitangents, normals, and UVs. Changing triangulation, tangent generation, or mirrored UV handling can change the result even when the image file is identical.

2

Y+ and Y- are conventions

OpenGL-style normals encode upward tangent Y as positive green; DirectX-style maps invert that component. A wrong convention makes dents read as ridges on one axis. Test by flipping only green, never the full RGB image.

3

Height needs a reference level

A grayscale height image is meaningless without a scale and midpoint. Record whether black is low, white is high, the physical displacement range, and whether 0.5 is the zero plane. Avoid stretching 8-bit data across large displacements.

4

Normal strength has limits

Excessive strength pushes tangent normals toward grazing directions and creates crunchy, unstable highlights. Use geometry for silhouette and large forms, displacement for true relief, and normals for sub-pixel or small-scale response.

Production workflow

1. Lock UVs, triangulation, and tangent-generation rules.
2. Keep a high-precision height master when the source supports it.
3. Generate a tangent normal at the intended texel density.
4. Confirm Y convention with a one-channel green flip test.
5. Preview on curved and flat geometry under a rotating light.
6. Inspect mips for renormalization and compression artifacts.

PLAYTEX AI ROUTE

PBR Map Generator

Derive a coherent normal and height pair from a prepared source.

playtex.ai/pbr-map-generator

Verify before approval

PASS CHECKS

- [ ] Does a known raised mark light from the correct side?
- [ ] Do UV seams stay stable on the final mesh triangulation?
- [ ] Is displacement scale stated in scene units?
- [ ] Does the map remain calm at distance and in motion?

COMMON FAILURE MODES

- Calling a height image a normal map because both create bumps.
- Flipping the whole normal map instead of the green channel.
- Using normal detail to fake a silhouette-scale form.

TWO-MINUTE CONTROL

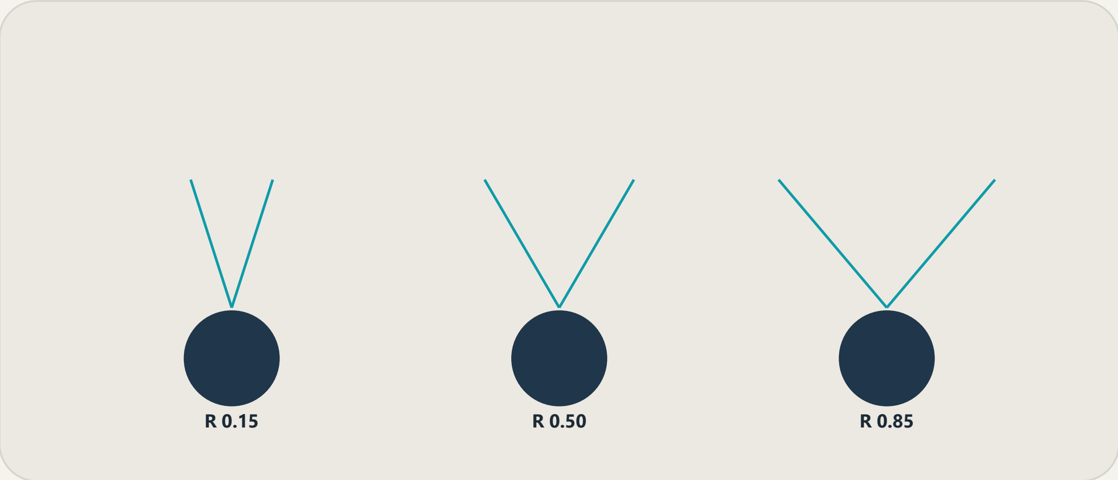
Flip only the green channel. If the perceived relief reverses correctly, the Y convention was wrong.

CHAPTER 05

Roughness, smoothness, and reflection spread

Read the highlight, not the grayscale thumbnail.

Roughness describes micro-surface variation and broadens reflections as it increases. Smoothness is usually the inverse control: smoothness = 1 - roughness. The destination shader decides which semantic it expects.



KEEP THIS

Roughness is easiest to calibrate by observing the size, clarity, and continuity of reflected shapes under a known environment.

What to understand

1

Variation needs material logic

Scratches may become rougher or smoother depending on whether they cut a polished surface, expose a substrate, hold moisture, or receive a new coating. Do not derive roughness from luminance with a universal invert.

2

Gloss inversion is exact only by contract

If a gloss or smoothness map is explicitly defined as the inverse of perceptual roughness, invert it once in linear data. Do not invert after sRGB decoding mistakes, and do not assume every legacy gloss map used the same response curve.

3

Low roughness exposes every flaw

Mirror-like response reveals seams, tangent discontinuities, compression blocks, and weak environment lighting. Diagnose those systems separately before adding noise to hide them.

4

Use range before detail

First establish the plausible minimum, maximum, and median roughness for the material. Then add spatial variation at meaningful scales. High-frequency noise alone creates sparkle, not realism.

Production workflow

1. Set a uniform roughness and rotate a high-contrast environment.
2. Match the overall reflection spread before adding texture.
3. Add broad variation tied to coating, wear, moisture, or contamination.
4. Add restrained micro-variation at the final texel density.
5. Invert once if the target requires smoothness.
6. Review mip levels and motion for shimmer.

PLAYTEX AI ROUTE

PBR Engine Converter

Convert roughness and smoothness conventions during export.

playtex.ai/tools/pbr-engine-converter

Verify before approval

PASS CHECKS

- [ ] Can you explain every major roughness change physically?
- [ ] Is a wide highlight coming from roughness rather than a blurred normal?
- [ ] Has roughness-to-smoothness conversion happened exactly once?
- [ ] Does the surface avoid glitter at gameplay distance?

COMMON FAILURE MODES

- Generating roughness directly from base-color luminance.
- Using contrast to compensate for weak lighting.
- Treating roughness 1 as a surface with no reflection.

TWO-MINUTE CONTROL

Use a uniform 0.2, 0.5, then 0.8 and compare reflection spread before judging the textured map.

CHAPTER 06

Metallic classification and mixed surfaces

Classify the visible layer, not the object category.

The metallic map chooses between dielectric and conductor response at each texel. Most clean, opaque regions belong near 0 or 1; intermediate pixels are most defensible at antialiased boundaries, blends, thin contamination, or limited-resolution transitions.

KEEP THIS

A steel object covered by paint is dielectric where paint is visible and metallic only where conductive metal is exposed.

What to understand

1

Material layers decide the value

Rust, oxide, dust, mud, paint, varnish, and fingerprints are not metal merely because they sit on metal. Classify the outermost optically dominant layer represented by the texel.

2

Gray is evidence of mixing

Gray metallic values represent a blend inside a filtered texel or an artistic approximation to layered coverage. Widespread arbitrary gray often creates dark, muddy energy balance and weakens portability.

3

Edges need filtering room

A hard binary mask still creates intermediate samples after texture filtering and mip generation. Give material boundaries enough resolution, consistent dilation, and stable UVs so the filtered transition stays localized.

4

Metal color belongs in base color

For a conductor, use the base-color RGB to represent colored reflectance. Do not tint the metallic map. Metallic is a scalar classification in the common workflow.

Production workflow

1. List every visible substance and coating in the material.
2. Assign clean dielectric regions to 0 and clean conductors to 1.
3. Build boundary masks from real layer coverage.
4. Inspect each mip to catch gray spreading across small islands.
5. Review metals under an environment with readable reflections.
6. Check that damaged coatings reveal the correct substrate stack.

PLAYTEX AI ROUTE

PLAYTEX AI Material Preflight

Flag ambiguous metallic ranges and channel mistakes.

playtex.ai/tools/material-preflight

Verify before approval

PASS CHECKS

- [ ] Are paint, rust, dust, and oxide classified as dielectrics?
- [ ] Is gray confined to justified transitions or partial coverage?
- [ ] Does exposed metal use plausible base-color reflectance?
- [ ] Do small metallic features survive downsampling without contaminating neighbors?

COMMON FAILURE MODES

- Using metallic to make plastic shinier.
- Filling a mixed asset with mid-gray.
- Calling every part of a metal object metallic.

TWO-MINUTE CONTROL

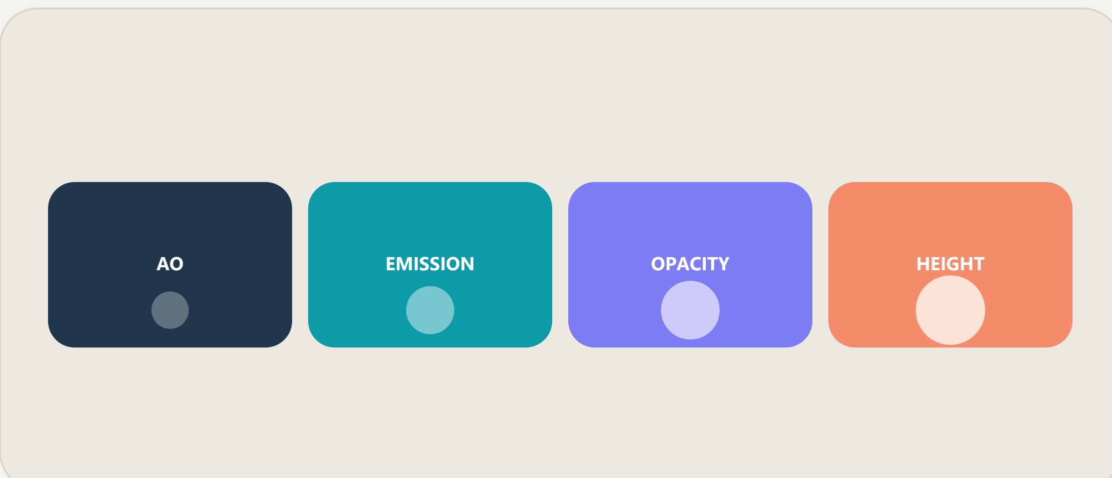
Force clean metal to 1 and coatings to 0. If response improves, the classification mask was ambiguous.

CHAPTER 07

AO, emission, opacity, and displacement

Support maps are powerful when their scope is explicit.

Ambient occlusion modulates indirect light, emission adds radiance-like surface output, opacity defines coverage or blending, and displacement changes geometry or its apparent position. None of these is a substitute for clean base color.



KEEP THIS

Treat each support map according to what the target shader actually does with it, especially for AO scope and transparency mode.

What to understand

1

AO is not a shadow photograph

Baked AO estimates local occlusion for ambient or indirect light. It should not lock direct-light shadows into the surface. Keep it separate and tune its influence in the material when possible.

2

Emission needs intensity context

An emissive texture often supplies color while a scalar supplies intensity. Values above display white may require HDR-capable parameters or formats. A glowing-looking bitmap does not automatically illuminate nearby geometry.

3

Coverage and transmission differ

Use cutout or mask for binary coverage such as leaves and fences; use blend for partial coverage such as smoke or gauze; use a transmission model for solid glass when supported. These modes solve different physical and sorting problems.

4

Displacement needs geometry

Vertex displacement can only move existing vertices. Parallax changes the apparent surface without changing silhouette. Tessellation or virtualized geometry can add detail but has engine and platform costs.

Production workflow

1. Choose the target shader and transparency mode first.
2. Keep AO, emission, opacity, and height as separate masters.
3. Define neutral values and scale for each map.
4. Test alpha edges over light and dark backgrounds.
5. Test displacement at the final mesh density and silhouette.
6. Pack channels only after behavior is approved.

PLAYTEX AI ROUTE

Alpha Mode Inspector

Choose and validate coverage behavior before export.

playtex.ai/tools/alpha-mode-inspector

Verify before approval

PASS CHECKS

- [ ] Does AO affect indirect response rather than painting permanent dirt?
- [ ] Is emissive intensity controlled independently from color where needed?
- [ ] Does alpha mode match the physical coverage problem?
- [ ] Is geometry dense enough for the displacement method?

COMMON FAILURE MODES

- Using blended transparency for every cutout asset.
- Expecting an emissive map to cast light automatically.
- Applying displacement to a mesh with too few vertices.

TWO-MINUTE CONTROL

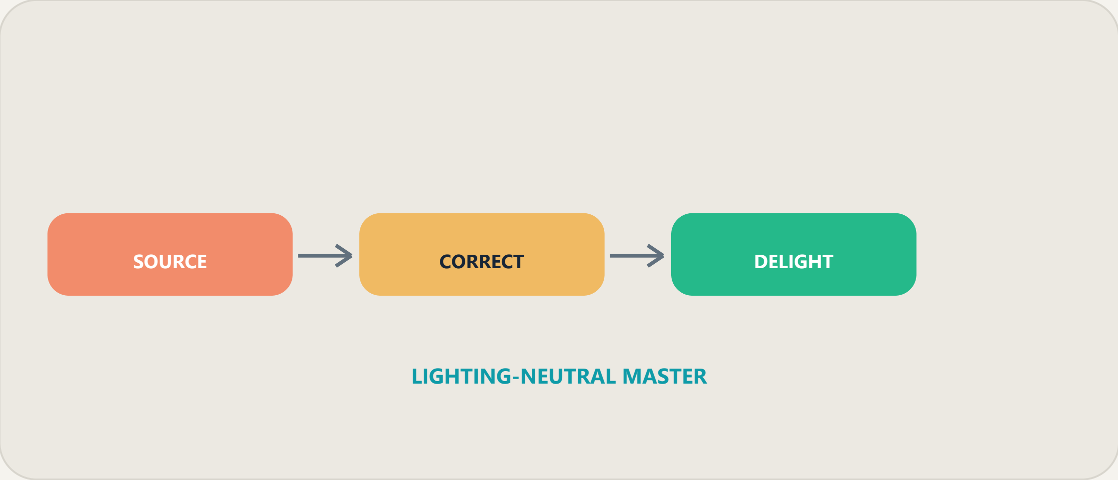
Disable AO, emission, alpha, and displacement one at a time to identify which support map drives the symptom.

CHAPTER 08

Source-image preparation and delighting

Fix the evidence before deriving the maps.

Source preparation converts a photograph, scan, render, or generated image into a stable material observation. It includes lens correction, perspective control, exposure and white-balance normalization, cleanup, scale definition, and delighting.



KEEP THIS

Map generation magnifies source errors. Remove lighting gradients and capture artifacts before extracting height, normal, or roughness.

What to understand

1

Capture for reconstruction

Use diffuse, even illumination; avoid clipped highlights and shadows; include a scale reference; keep the camera as orthogonal as practical; and capture enough overlap or coverage to replace damaged regions.

2

Correct optics before tiling

Lens distortion, perspective convergence, chromatic aberration, and vignetting become repeating patterns. Correct them before crop and tile construction so borders represent the same scale and projection.

3

Delight in layers

Remove broad gradients first, then localized cast shadows, then specular contamination. Preserve real pigment and embedded material variation. Compare against the source frequently to avoid sanding away identity.

4

Generated inputs still need provenance

AI-generated or procedural sources can contain inconsistent lighting cues and impossible microstructure. Record the prompt or graph, intended scale, edit history, and license, then apply the same neutral-source tests as a photograph.

Production workflow

1. Archive the untouched source and record capture conditions.
2. Correct lens, perspective, white balance, and exposure.
3. Remove unwanted objects and sensor artifacts.
4. Remove broad illumination, then local highlights and shadows.
5. Define real-world scale and a representative crop.
6. Approve the cleaned base before generating dependent maps.

PLAYTEX AI ROUTE

Material Delighter / True Albedo Cleaner

Prepare a lighting-neutral base before downstream generation.

playtex.ai/tools/material-delighter

Verify before approval

PASS CHECKS

- [ ] Does rotating the tile leave no fixed light direction?
- [ ] Are repeated gradients absent at 2x2 and 4x4 views?
- [ ] Did delighting preserve pigment and material landmarks?
- [ ] Can the source and every edit be traced?

COMMON FAILURE MODES

- Cropping before correcting perspective.
- Removing real wear while removing highlights.
- Treating generated imagery as physically calibrated data.

TWO-MINUTE CONTROL

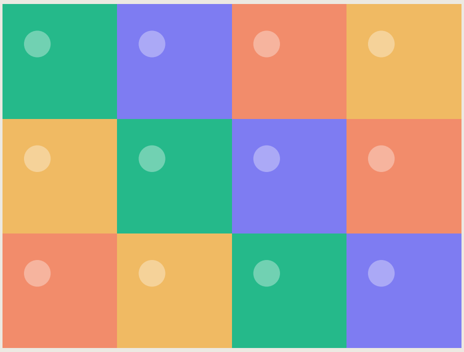
Rotate the source tile 180 degrees. A fixed light direction should become obvious immediately.

CHAPTER 09

Creating seamless materials

A hidden border is not the same as a convincing tile.

A seamless material must match at opposite edges and avoid recognizable repetition across larger fields. Continuity is a local problem; repetition is a distribution and scale problem.



CHECK 4 x 3, NOT ONLY ONE BORDER

KEEP THIS

Validate every dependent map as a synchronized set. A seamless base color with a discontinuous normal or roughness map is still a broken material.

What to understand

1

Offset exposes the seam

Shift the image by half its width and height so borders meet in the center. Repair the crossing with structure-aware cloning, patching, or procedural synthesis. Recheck original borders after every large edit.

2

Kill landmarks, keep character

Remove unique stains, stones, knots, or cracks that become obvious stamps, but preserve enough frequency variation to keep the material recognizable. Judge at the intended world scale.

3

Maps must share structure

Height edits should drive regenerated normals and compatible AO. If maps are repaired independently, cracks can produce color without depth or highlights that no longer align with the surface.

4

Variation belongs above the tile

Large-scale color, wetness, dirt, and macro breakup are often better added in the engine with masks, decals, second UVs, or world-space variation than baked into a small repeating source.

Production workflow

1. Set physical scale and crop a representative area.
2. Offset X and Y so all borders meet at center.
3. Repair structural crossings before fine texture.
4. Remove memorable landmarks and broad gradients.
5. Regenerate dependent maps from the corrected masters.
6. Inspect 2x2, 4x4, and gameplay-scale repetition.

PLAYTEX AI ROUTE

PBR Texture-Set Rotator

Rotate a whole map stack without breaking directional data.

playtex.ai/tools/pbr-texture-set-rotator

Verify before approval

PASS CHECKS

- [ ] Are borders continuous in every map and mip?
- [ ] Can you spot the tile origin in a 4x4 grid?
- [ ] Do features have a believable real-world size?
- [ ] Is macro variation separated from the reusable tile?

COMMON FAILURE MODES

- Blurring the seam until structure melts.
- Fixing each map independently.
- Using one tile to carry both micro and landscape-scale variation.

TWO-MINUTE CONTROL

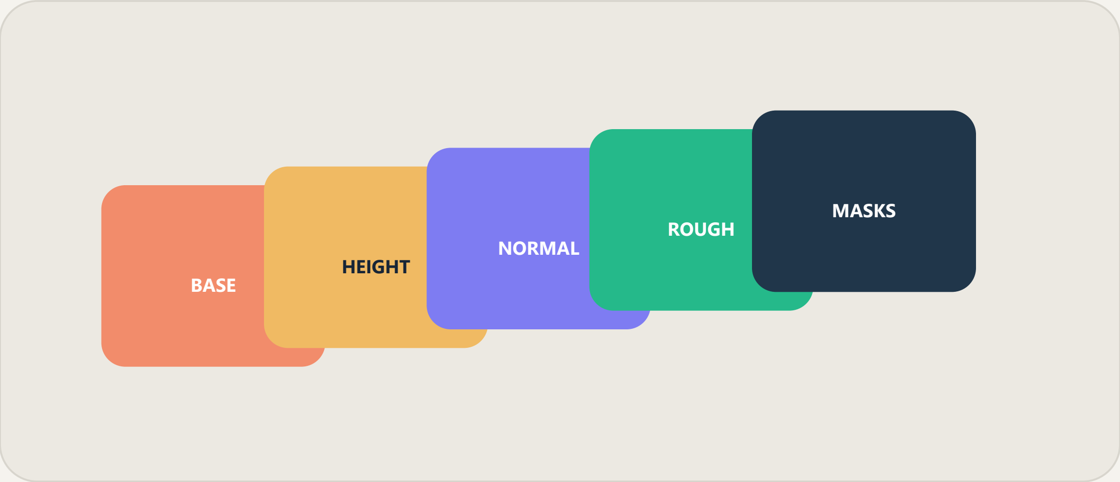
Show a 4x4 repeat with a temporary high-contrast border; verify every map shares the same tile origin.

CHAPTER 10

Building a coherent map stack

One material, one shared history.

A coherent stack is a set of maps that agree on UVs, dimensions, physical scale, feature placement, material boundaries, and version. Coherence matters more than the beauty of any single thumbnail.



KEEP THIS

Choose canonical masters, derive dependent maps, and export all targets from the same approved revision.

What to understand

1

Separate observation from derivation

Base color and height may come from cleaned observations. Normals can derive from height; AO may derive from geometry or height; masks may derive from classified layers. Document the graph so a change propagates intentionally.

2

Boundaries are shared assets

Paint edges, exposed substrate, wet zones, and cutout coverage often affect multiple maps. Store these boundaries as reusable masks rather than repainting similar edges in each channel.

3

Resample once, late

Repeated resizing softens masks and normals and changes binary coverage. Work at a chosen master resolution, then produce final sizes and mips in a controlled export stage.

4

Version the set, not just files

A folder containing base color v7, normal v5, and roughness final2 is not a reliable material. Use a set-level version or manifest that records source hashes, export preset, dimensions, and target.

Production workflow

1. Define a canonical resolution, UV set, and physical scale.
2. Create reusable masks for material boundaries and coverage.
3. Identify which maps are authored and which are derived.
4. Propagate structural edits through dependent maps.
5. Export all channels from one named set revision.
6. Preflight dimensions, modes, and bindings together.

PLAYTEX AI ROUTE

Texture Node Editor

Keep derivation and channel operations explicit and repeatable.

playtex.ai/tools/texture-node-editor

Verify before approval

PASS CHECKS

- [ ] Do the same features align across color, normal, roughness, and masks?
- [ ] Are all maps from the same set revision?
- [ ] Was each map resized only through the export pipeline?
- [ ] Can the full delivery be rebuilt from canonical sources?

COMMON FAILURE MODES

- Hand-fixing one exported map without updating its source.
- Mixing resolutions without a deliberate sampling reason.
- Using filenames as the only version system.

TWO-MINUTE CONTROL

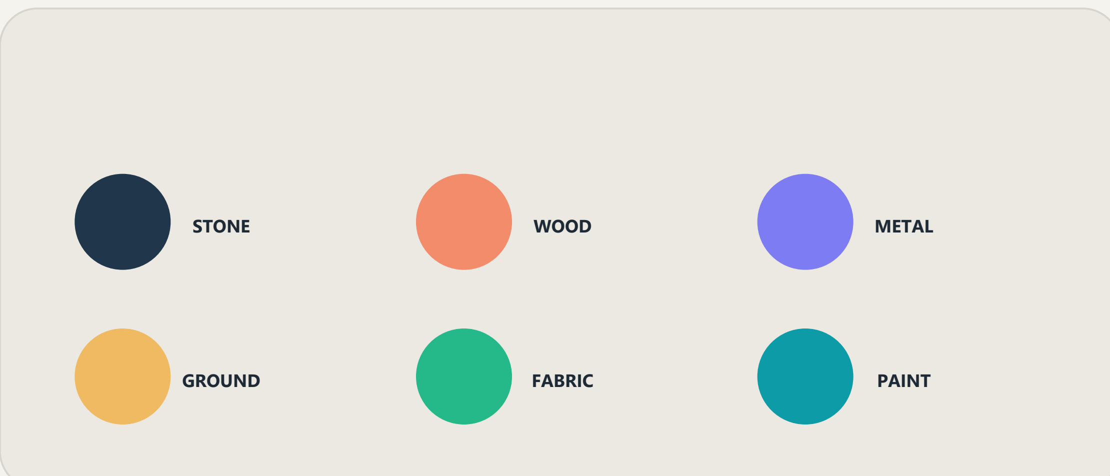
Overlay map edges at 50 percent opacity to reveal misalignment or stale revisions.

CHAPTER 11

Material calibration by surface family

Start from physical identity, then tune the finish.

Calibration is the process of setting plausible ranges and relationships for a named surface under controlled lighting. Stone, wood, metal, ground, fabric, and paint each fail in characteristic ways, but no universal slider value replaces observation.



KEEP THIS

The ranges in this handbook are review windows, not standards. Moisture, polish, wear, fiber direction, grain, coatings, and scale can move a real surface outside them.

What to understand

1

Stone and ground

Dry mineral surfaces are usually dielectric and broadly rough, with restrained base-color contrast. Wetness mainly lowers roughness, darkens apparent color, and can add a coherent film; it does not turn the ground metallic.

2

Wood and fabric

Both are dielectric. Wood response depends strongly on finish: unfinished fibers are rough, while varnish adds a smoother coat. Fabric is usually rough with directional fiber response that can require sheen or anisotropy rather than metallic values.

3

Exposed metal

Set clean conductive regions to metallic 1 and calibrate their base color from plausible reflectance. Roughness carries polish, brushing, oxidation, fingerprints, and wear. Oxide and dirt layers are dielectric.

4

Painted surfaces

Paint is a dielectric coating over a substrate. Intact paint stays metallic 0 even on steel; chips reveal metal locally. Clearcoat, orange peel, scratches, and grime belong to distinct scales and may need separate lobes or masks.

Production workflow

1. Name the substrate, coating, contaminants, and moisture state.
2. Set metallic classification from the visible top layer.
3. Choose a broad roughness window from the finish.
4. Match base color under neutral exposure.
5. Add structure at macro, meso, and micro scales.
6. Compare against reference across at least three lights.

PLAYTEX AI ROUTE

AI Texture Generator

Generate focused source variations, then calibrate and preflight them.

playtex.ai/ai-texture-generator

Verify before approval

PASS CHECKS

- [ ] Is the visible coating classified separately from its substrate?
- [ ] Does wetness affect several channels coherently?
- [ ] Are fiber, grain, brushing, and scratches aligned to real directions?
- [ ] Are numerical ranges treated as review prompts rather than truth?

COMMON FAILURE MODES

- Making all dark materials metallic.
- Using the same roughness noise for every surface family.
- Ignoring coatings and moisture state.

TWO-MINUTE CONTROL

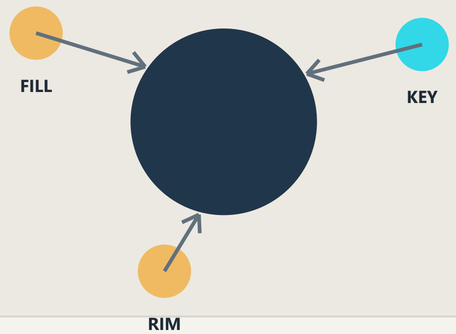
Swap to a neutral sphere and compare a uniform-value sweep before adding micro-variation.

CHAPTER 12

Reviewing maps under controlled lighting

A repeatable review rig turns taste into evidence.

Controlled review uses fixed geometry, exposure, camera, tone mapping, and a small set of known lights so changes in the material remain visible. Creative lighting comes after the material passes this neutral test.



KEEP THIS

Use multiple simple rigs because each reveals a different failure: neutral environment for balance, hard key for normals, grazing reflection for roughness, and unlit view for base color.

What to understand

1

Use diagnostic geometry

A sphere reveals highlight shape and grazing response; a plane reveals tiling and tangent direction; a rounded cube exposes seams and edge behavior. Review on the final asset too, but do not rely on it alone.

2

Lock the camera pipeline

Fixed exposure, tone mapper, white point, and output transform make comparisons meaningful. Automatic exposure can make two materially different surfaces appear similar by compensating between captures.

3

Rotate light and environment

A static light can hide inverted normals or discontinuous roughness. Rotate a hard key and a high-contrast environment through known angles and watch whether surface response moves smoothly.

4

Compare one variable at a time

Use A/B captures with identical settings and change only one map, channel, or scalar. Save the rig with the asset review preset so future revisions use the same evidence.

Production workflow

1. Load a sphere, plane, rounded cube, and the production mesh.
2. Lock exposure, white balance, tone mapping, and camera.
3. Review unlit base color and isolated data maps.
4. Run neutral, hard-key, grazing, and alternate-environment tests.
5. Capture labeled A/B frames at identical settings.
6. Approve only after the material survives motion and distance.

PLAYTEX AI ROUTE

AI HDRI Generator

Build contrasting review environments without changing the material.

playtex.ai/hdri-sphere-generator

Verify before approval

PASS CHECKS

- [ ] Can another reviewer reproduce the same view?
- [ ] Is auto exposure disabled or recorded?
- [ ] Do normal and roughness features respond continuously as light moves?
- [ ] Has the material been checked both close and at gameplay distance?

COMMON FAILURE MODES

- Approving only a dramatic hero render.
- Changing the light while comparing material revisions.
- Ignoring tone mapping and exposure differences.

TWO-MINUTE CONTROL

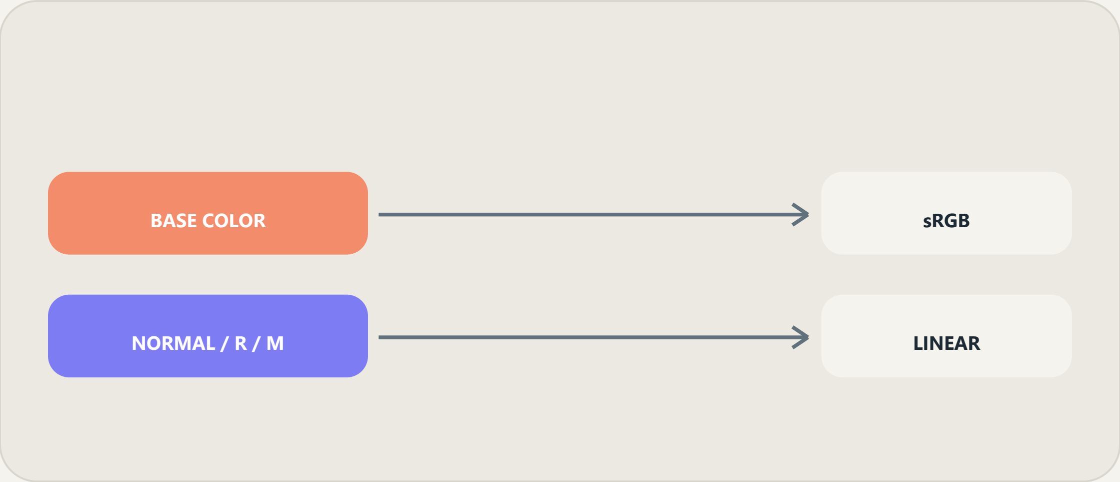
Capture A/B frames with locked exposure and only one changed variable.

CHAPTER 13

Color space, bit depth, naming, and file formats

Storage choices are part of the material contract.

A texture's pixel values are meaningful only when the sampler knows whether they represent display-oriented color or linear data. Bit depth controls precision; file format controls features and interchange; naming carries semantics into production.



KEEP THIS

Base color and common emissive color are usually sRGB-encoded. Normal, roughness, metallic, AO, height, and packed masks are usually sampled as linear or non-color data.

What to understand

1

Color and data take different paths

Lighting math operates in linear space. Color textures commonly use sRGB encoding for efficient storage and are decoded on sample. Numeric data must bypass that transfer function or midrange values will be altered.

2

Use precision where operations need it

Eight-bit output is often sufficient for final color and many masks, but gradients, height, displacement, aggressive levels, and repeated transforms can band. Keep higher-precision masters and quantize once at delivery.

3

Choose formats by job

PNG is lossless and supports alpha and up to 16-bit samples. OpenEXR supports half and float channels for HDR or precision-heavy data. JPEG is lossy and has no alpha; avoid it for normals, packed masks, or critical boundaries.

4

Names should survive context loss

A useful filename states asset, material, map semantic, optional packing, resolution or variant, target, and version. Do not use Diffuse when the shader expects metallic-roughness base color unless that distinction is deliberate.

Production workflow

1. Label every master as color-bearing or numeric data.
2. Choose working precision from the strongest edit and gradient need.
3. Select an interchange format that preserves required channels.
4. Apply one project naming grammar across the set.
5. Quantize and compress only in the target export stage.
6. Verify importer sampling rather than trusting the extension.

PLAYTEX AI ROUTE

PLAYTEX AI Material Preflight

Inspect format, channels, dimensions, and likely sampling risks.

playtex.ai/tools/material-preflight

Verify before approval

PASS CHECKS

- [ ] Are data maps bypassing sRGB conversion?
- [ ] Do height gradients survive at the chosen bit depth?
- [ ] Does the format preserve alpha and channel precision?
- [ ] Can the filename identify both semantic and target?

COMMON FAILURE MODES

- Assuming PNG metadata overrides a shader's declared semantics.
- Using JPEG for normals or packed masks.
- Encoding version history as final_final2.

TWO-MINUTE CONTROL

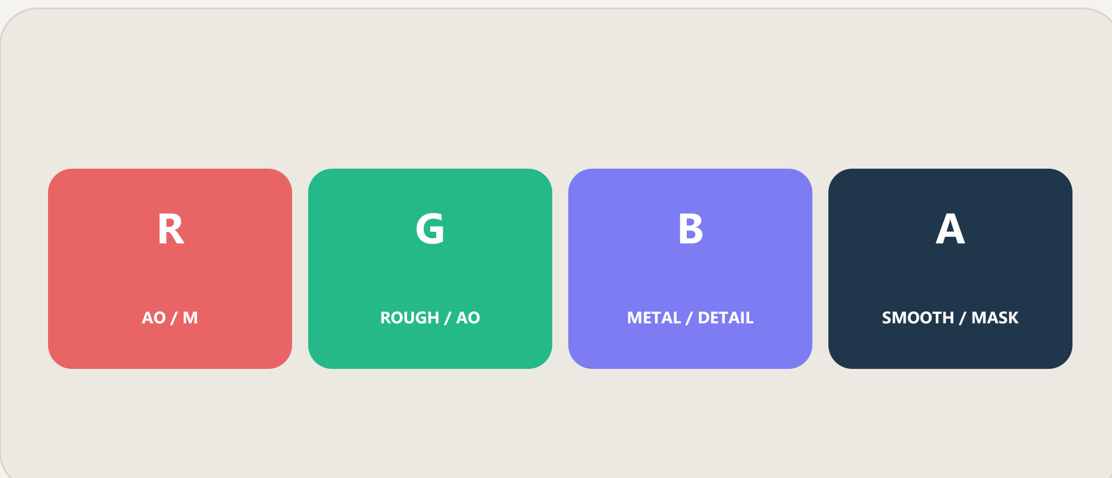
Toggle sRGB on one suspect data map and watch whether the midrange behavior changes.

CHAPTER 14

Channel packing and export conventions

Packing is a delivery format, not a source of truth.

Channel packing stores independent grayscale maps in R, G, B, or A of one texture. It can reduce texture samples and files, but the order is renderer-specific and the packed texture must be sampled as non-color data.



KEEP THIS

Keep unpacked canonical maps. Generate named target packs through presets, and verify every channel in isolation after import.

What to understand

1

Common packs are not universal

glTF and Godot commonly use ORM: occlusion in R, roughness in G, metallic in B. Unreal projects often choose project-defined orders such as ORM. Unity URP and Built-In commonly use metallic in R and smoothness in A; HDRP mask maps use metallic R, AO G, detail mask B, smoothness A.

2

Alpha can change compression

Adding an alpha channel may select a different GPU compression format and increase memory. Pack based on the target platform and quality needs, not only file count.

3

Correlated artifacts matter

Block compression treats neighboring channels within the same texture format. Fine roughness, hard metallic boundaries, and smooth AO gradients may have different quality needs. Inspect the final GPU format, not just the source PNG.

4

Write the order into the name

Use an explicit token such as _ORM, _MRA, or _MaskMap, plus a target token when necessary. A generic _Packed filename forces every downstream user to guess.

Production workflow

1. Approve unpacked source maps first.
2. Select a documented preset for the target shader.
3. Invert roughness only if the destination expects smoothness.
4. Pack exact channels without sRGB transforms.
5. Name the channel order and target explicitly.
6. Inspect channels and GPU-compressed results after import.

PLAYTEX AI ROUTE

PBR Engine Converter

Create documented target packs from canonical maps.

playtex.ai/tools/pbr-engine-converter

Verify before approval

PASS CHECKS

- [ ] Does the filename declare the packing order?
- [ ] Is the packed image sampled as linear data?
- [ ] Has roughness/smoothness inversion occurred exactly once?
- [ ] Does alpha justify its memory and compression cost?

COMMON FAILURE MODES

- Treating ORM as a universal engine standard.
- Color-managing a packed mask texture.
- Discarding the separate masters after packing.

TWO-MINUTE CONTROL

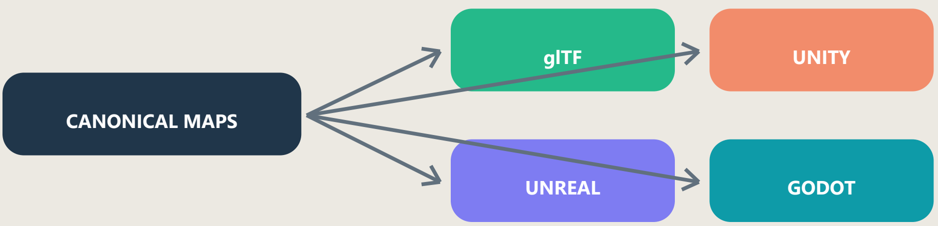
Display R, G, B, and A separately after import using known constant values.

CHAPTER 15

Engine-import matrix

The same pixels can mean different things at the destination.

An engine-import matrix records how each renderer expects map semantics, channel order, color-space flags, normal convention, and alpha mode. It is a release artifact that should be versioned with export presets.



KEEP THIS

Do not memorize one universal setup. Identify the shader and pipeline, apply its contract, then run a known-value import test.

What to understand

1

Shader choice comes first

Unity Built-In, URP, and HDRP do not share one identical pack. Unreal material graphs can use any documented project order. Godot ORMMaterial3D and glTF have defined ORM layouts. Blender and Three.js depend on explicit node or property wiring.

2

Defaults can evolve

Importers may detect normals, change compression, or assign sRGB automatically. Record engine version, rendering pipeline, shader name, preset version, and any importer overrides so a future upgrade can be audited.

3

Use a known-value probe

Build a 4x4 test image with distinct channel values and a normal patch with a known raised direction. Import it before a large delivery to reveal swapped channels, hidden inversion, and color-space conversion immediately.

4

Round-trip only when necessary

Exporting from one engine and re-importing into another can compound packing and conversion. Return to canonical sources and generate a new target export whenever possible.

Production workflow

1. Choose engine, render pipeline, shader, and version.
2. Select the matching packing and normal preset.
3. Set color or non-color sampling for every texture.
4. Import a known-value probe and inspect isolated channels.
5. Bind the production set and compare with the canonical preview.
6. Save the verified preset and matrix beside the delivery.

PLAYTEX AI ROUTE

PBR Engine Converter

Generate engine-specific outputs from a declared target.

playtex.ai/tools/pbr-engine-converter

Verify before approval

PASS CHECKS

- [ ] Does the matrix name an exact shader or material model?
- [ ] Are normal Y and tangent assumptions documented?
- [ ] Did an actual import test confirm each channel?
- [ ] Can a future engine upgrade be compared against the preset?

COMMON FAILURE MODES

- Writing Unity or Unreal without naming the pipeline or shader.
- Trusting automatic detection without inspecting the import.
- Converting from another delivery file instead of masters.

TWO-MINUTE CONTROL

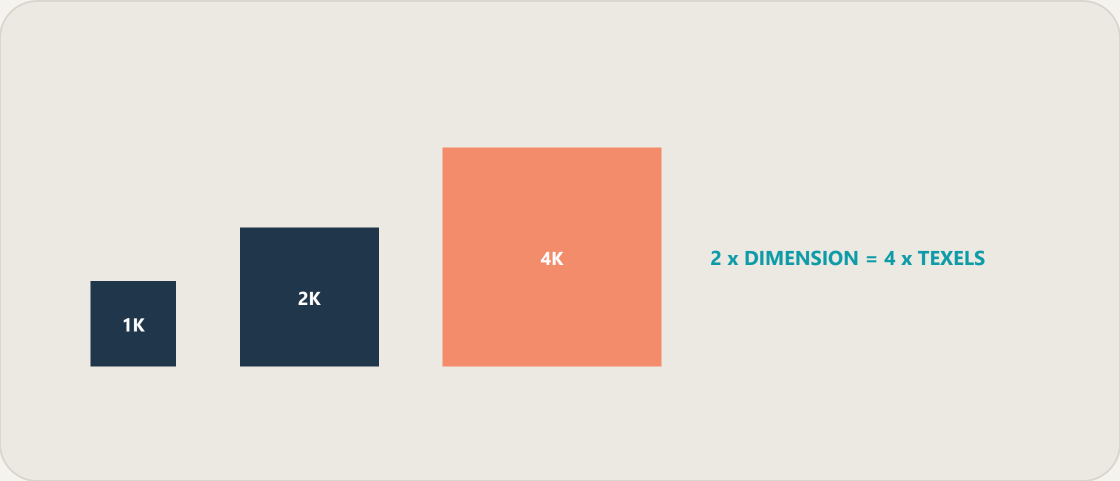
Import a four-channel probe before the production set and verify each destination input.

CHAPTER 16

Performance and texture-budget fundamentals

Budget runtime memory and sampling, not just download size.

Texture cost includes GPU memory, bandwidth, sampling, streaming, disk or network payload, and CPU-side management. A small PNG on disk may become a large uncompressed or GPU-compressed resource at runtime.



KEEP THIS

Start with screen-space need and platform constraints. Resolution, format, mip residency, array or atlas strategy, and shader samples all affect the actual budget.

What to understand

1

Resolution grows quadratically

Doubling width and height quadruples base-level texels. An uncompressed 4096 RGBA8 texture is 64 MiB before mips; a full mip pyramid adds about one third for conventional 2D power-of-two textures.

2

Compression is semantic

BC1 can suit opaque color with limited quality needs; BC5 is designed for two-component data such as normals; BC7 offers higher-quality color at a larger block size. Mobile and web targets may use ASTC, ETC, or transcodable KTX2 workflows.

3

Mips improve stability

Mipmaps reduce minification aliasing and allow streamers to load an appropriate resolution. Removing mips to save memory often trades a predictable overhead for shimmer, cache pressure, and unstable motion.

4

Measure the resident set

The useful number is not the sum of source-file sizes. Profile what is resident in representative scenes, include UI and environment textures, observe pool pressure, and test target hardware.

Production workflow

1. Define target hardware and a scene-level texture budget.
2. Estimate texel density from closest intended viewing distance.
3. Choose dimensions per asset role, not a universal 4K default.
4. Select GPU formats by semantic and platform.
5. Enable and inspect mip generation and streaming.
6. Profile representative scenes on target hardware.

PLAYTEX AI ROUTE

Texture Budget & VRAM Analyzer

Estimate base levels, mip overhead, and set-level memory.

playtex.ai/tools/texture-memory-calculator

Verify before approval

PASS CHECKS

- [ ] Is the estimate based on runtime format rather than PNG size?
- [ ] Are full mip chains included in memory estimates?
- [ ] Does each high-resolution texture earn its screen-space detail?
- [ ] Has the worst representative scene been profiled on target?

COMMON FAILURE MODES

- Treating disk size as VRAM usage.
- Making every map the same resolution.
- Disabling mipmaps to sharpen a still image.

TWO-MINUTE CONTROL

Temporarily cap maximum size one mip lower and compare visible quality against the memory saved.

CHAPTER 17

Production QA checklist

Quality is a gate, not a final glance.

Production QA verifies the source stack, semantic correctness, visual behavior, engine import, runtime stability, and delivery package. Each gate should produce evidence that another person can reproduce.

- [x] **SOURCE**
- [x] **MAPS**
- [x] **IMPORT**
- [ ] **RUNTIME**

KEEP THIS

Automate objective checks such as dimensions, modes, channels, names, and links. Reserve human review for physical plausibility, visual continuity, and target-context judgment.

What to understand

1

Source gate

Confirm provenance, scale, UVs, canonical masters, layer masks, bit depth, and version. Reject baked lighting, clipped data, or unexplained edits before export.

2

Map gate

Review each channel alone: semantic, range, neutral value, alignment, seams, edge dilation, normal convention, color-space role, and correlation with the other maps.

3

Engine gate

Import with the target preset, inspect bindings and sampler flags, review known-value probes, compare against the canonical preview, and capture the exact shader and engine version.

4

Runtime and delivery gate

Check mips, streaming, compression, motion, memory, naming, manifest, licenses, and package integrity. Reopen the delivered files rather than trusting the working folder.

Production workflow

1. Freeze a candidate set revision and generate a manifest.
2. Run automated structural and semantic preflight.
3. Review maps individually and as a bound stack.
4. Test controlled lights, distance, motion, and backgrounds.
5. Profile the target build and capture evidence.
6. Package only the verified outputs and documentation.

PLAYTEX AI ROUTE

PLAYTEX AI Material Preflight

Run a repeatable structural check before human review.

playtex.ai/tools/material-preflight

Verify before approval

PASS CHECKS

- [ ] Does every pass criterion have an owner and evidence?
- [ ] Was the delivered package reopened and rechecked?
- [ ] Are engine and preset versions recorded?
- [ ] Can the set be regenerated without manual hidden steps?

COMMON FAILURE MODES

- Letting a hero render substitute for channel inspection.
- Running QA on sources but shipping different exports.
- Keeping exceptions only in chat or memory.

TWO-MINUTE CONTROL

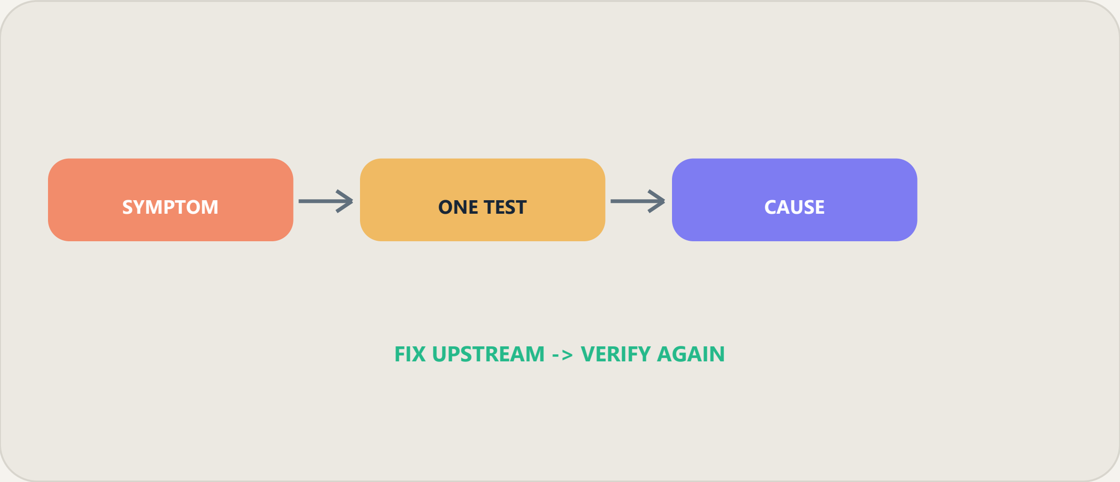
Reopen the packaged delivery in a clean project rather than the authoring project.

CHAPTER 18

Troubleshooting reference

Change one variable, run one decisive test.

Texture troubleshooting is fastest when you start from a visible symptom, form a narrow hypothesis, and use a reversible test. Random slider changes destroy evidence and often create a second problem.



KEEP THIS

Isolate source data, import interpretation, shader binding, lighting, and runtime delivery as separate layers. Test the cheapest likely layer first.

What to understand

1

Wrong shape response

If dents look raised, flip only normal green as a two-minute test. If seams remain, check tangent basis, UV mirroring, triangulation, and compression. If the silhouette is wrong, normals cannot solve it.

2

Wrong reflectance

If everything looks plastic, neutralize base color, set metallic from classification, sweep roughness uniformly, and use a readable environment. Reintroduce textured variation one map at a time.

3

Wrong color or masks

If midtones shift, inspect sRGB flags. If material regions swap behavior, isolate packed channels with constant colors. If transparency halos appear, inspect edge RGB, alpha mode, and premultiplication expectations.

4

Wrong distance behavior

If the material shimmers or blurs, inspect mip generation, normal renormalization, anisotropy, compression, texel density, and streaming residency. Do not disable mips as the default fix.

Production workflow

1. Record the symptom, view, light, distance, and target build.
2. Replace maps with known constants to isolate the failing input.
3. Run one reversible two-minute test.
4. Apply the correction at the canonical source or preset layer.
5. Retest the full stack and at least one alternate environment.
6. Add the failure and prevention rule to the project checklist.

PLAYTEX AI ROUTE

PLAYTEX AI Material Preflight

Start with a structural baseline before visual diagnosis.

playtex.ai/tools/material-preflight

Verify before approval

PASS CHECKS

- [ ] Did the test isolate one layer rather than change several?
- [ ] Was the fix applied upstream so exports remain reproducible?
- [ ] Did the correction survive motion, mips, and another light?
- [ ] Is the prevention rule now documented?

COMMON FAILURE MODES

- Compensating for a wrong map with lighting or exposure.
- Making destructive edits directly in an engine import.
- Stopping after the first flattering view.

TWO-MINUTE CONTROL

Replace the suspect map with its neutral constant and change nothing else.

APPENDICES

Production sheets for calibration, naming, engine import, memory, QA, and troubleshooting.

CALIBRATION

NAMING

ENGINE MATRIX

MEMORY

QA

TROUBLESHOOTING

Use these sheets as living project documents. Revalidate after pipeline upgrades.

Material calibration cards

Stone

DIELECTRIC; METALLIC 0

REVIEW STARTING WINDOW - NOT A STANDARD

Dry roughness often 0.65-0.90

READ THESE SIGNALS

Mineral grain, fractures, pores, polish, dust, and wet films should affect maps at matching scales.

TYPICAL FAILURE

Over-dark albedo, cavity AO baked into color, or uniform procedural noise makes stone read as rubber.

Wood

DIELECTRIC; METALLIC 0

REVIEW STARTING WINDOW - NOT A STANDARD

Unfinished roughness often 0.50-0.80; coated often 0.15-0.45

READ THESE SIGNALS

Grain direction, pores, end grain, finish thickness, scratches, and moisture organize the response.

TYPICAL FAILURE

A single isotropic roughness layer can make every finish look like plastic laminate.

Material calibration cards

Metal

CLEAN EXPOSED CONDUCTOR NEAR METALLIC 1

REVIEW STARTING WINDOW - NOT A STANDARD

Roughness can span roughly 0.15-0.75 by polish and oxidation

READ THESE SIGNALS

Base-color reflectance, brushing direction, fingerprints, oxide, edge wear, and coating boundaries matter.

TYPICAL FAILURE

Gray metallic everywhere produces muddy hybrid response; rust and paint usually belong at metallic 0.

Ground

DIELECTRIC MIXTURE; METALLIC 0

REVIEW STARTING WINDOW - NOT A STANDARD

Dry roughness often 0.65-1.00; wet films often 0.15-0.45

READ THESE SIGNALS

Compaction, particle scale, moisture, footprints, puddles, and embedded debris need coherent masks.

TYPICAL FAILURE

A wet ground edit that only darkens color looks stained rather than wet.

Material calibration cards

Fabric

DIELECTRIC; METALLIC 0

REVIEW STARTING WINDOW - NOT A STANDARD

Base roughness often 0.65-1.00

READ THESE SIGNALS

Weave direction, fiber scale, fuzz, sheen, compression, and wear can require directional response.

TYPICAL FAILURE

High-contrast normal noise creates glitter and moire instead of readable fibers.

Painted surface

PAINT METALLIC 0; EXPOSED METAL LOCALLY 1

REVIEW STARTING WINDOW - NOT A STANDARD

Roughness often 0.30-0.80; clearcoat may be lower

READ THESE SIGNALS

Primer, paint, clearcoat, orange peel, chips, scratches, dirt, and substrate form a layer stack.

TYPICAL FAILURE

Marking intact automotive or industrial paint as metallic confuses coating and substrate.

Naming convention sheet

asset_material_semantic_precision_size_target_version.ext

BC	**Base color** cliff_granite_BC_2K_v003.png	Color / sRGB
NGL	**Normal OpenGL Y+** cliff_granite_NGL_2K_v003.png	Data / linear
NDX	**Normal DirectX Y-** cliff_granite_NDX_2K_v003.png	Data / linear
R	**Roughness** cliff_granite_R_2K_v003.png	Data / linear
S	**Smoothness** cliff_granite_S_2K_UnityURP_v003.png	Data / linear
M	**Metallic** cliff_granite_M_2K_v003.png	Data / linear

Naming convention sheet

Use a target token only when the pixels or packing differ by destination. Keep source names engine-neutral.

AO	**Ambient occlusion**	cliff_granite_AO_2K_v003.png	**Data / linear**
H	**Height**	cliff_granite_H_16b_2K_v003.png	**Data / linear**
E	**Emission**	panel_warning_E_2K_v003.png	**Usually color / sRGB**
O	**Opacity**	leaf_oak_O_1K_v003.png	**Coverage / linear**
ORM	**glTF or Godot pack**	cliff_granite_ORM_2K_glTF_v003.png	**R=AO G=R B=M**
MASKMAP	**Unity HDRP pack**	panel_paint_MaskMap_2K_HDRP_v003.png	**R=M G=AO B=Detail A=S**

Avoid: diffuse unless the workflow truly means diffuse; packed without a channel order; normal without Y convention when variants coexist; and final, final2, newest, or fix as version tokens.

Engine import matrix

glTF 2.0 metallic-roughness

BASE / COLOR	baseColor RGB: sRGB; alpha: linear coverage; straight/unassociated
DATA / PACK	ORM: R occlusion, G roughness, B metallic; linear
NORMAL	Tangent-space +Y; linear; alpha ignored
ALPHA	OPAQUE, MASK, or BLEND via material.alphaMode

Use MikkTSpace-compatible tangents when generated. Verify extensions before relying on clearcoat, transmission, or emissive strength.

Unity Built-In Standard

BASE / COLOR	Albedo/base color as color texture
DATA / PACK	Metallic R; smoothness commonly metallic A or albedo A
NORMAL	Import as Normal map; Flip Green Channel when source convention differs
ALPHA	Opaque, Cutout, Fade, or Transparent rendering mode

Built-In is deprecated in current Unity documentation; keep pipeline named in the preset.

Engine import matrix

Unity URP Lit

BASE / COLOR	Base Map as color texture
DATA / PACK	Common Lit packing: metallic R, occlusion G, smoothness A; B unused
NORMAL	Import as Normal map; confirm green convention
ALPHA	Surface Type plus Alpha Clipping and blend settings

Name the exact URP shader. Custom Shader Graphs can use different layouts.

Unity HDRP Lit

BASE / COLOR	Base Color Map as color texture
DATA / PACK	Mask Map: R metallic, G AO, B detail mask, A smoothness; linear
NORMAL	Tangent or object-space options depend on material setup
ALPHA	Surface Type, blend mode, and alpha clipping settings

Do not substitute a glTF ORM image for an HDRP Mask Map without repacking and roughness inversion.

Engine import matrix

Unreal Engine material

BASE / COLOR	Base Color sampled as color; remove captured lighting
DATA / PACK	Roughness, Metallic, AO, and packed masks: sRGB off; channel order is project-defined
NORMAL	Use Normal compression/sampler; flip green if the source convention differs
ALPHA	Blend Mode and Opacity or Opacity Mask input

Many teams use ORM, but the material graph is the contract. Record it explicitly.

Godot ORMMaterial3D

BASE / COLOR	Albedo as color texture
DATA / PACK	ORM: R occlusion, G roughness, B metallic
NORMAL	OpenGL-style +Y; Godot can invert green at import
ALPHA	Transparency mode on BaseMaterial3D

Godot uses normal R and G for compression; inspect import hints and final VRAM format.

Engine import matrix

Blender Principled BSDF

BASE / COLOR	Image Texture to Base Color using color interpretation
DATA / PACK	Roughness, Metallic, AO masks, and packed channels set to Non-Color
NORMAL	Image Texture Non-Color -> Normal Map node -> Principled Normal
ALPHA	Image alpha to Principled Alpha plus material blend settings

For glTF export, follow the exporter-supported node patterns and +Y tangent normal convention.

Three.js MeshStandardMaterial

BASE / COLOR	map uses SRGBColorSpace
DATA / PACK	aoMap R, roughnessMap G, metalnessMap B; NoColorSpace
NORMAL	Tangent-space normalMap; NoColorSpace; normalScale.y can compensate handedness
ALPHA	transparent, alphaTest, alphaMap, and blending settings

aoMap requires the expected UV channel. Use an environment map for meaningful PBR review.

Texture memory estimator

BASE BYTES = WIDTH x HEIGHT x BYTES PER PIXEL

FULL MIP CHAIN APPROX. = BASE x 4 / 3

SIZE	RGBA8 BASE	RGBA8 + MIPS	BC1 + MIPS	BC5/7 + MIPS
1024 x 1024	4 MiB	5.33 MiB	0.50 MiB	1.00 MiB
2048 x 2048	16 MiB	21.33 MiB	2.00 MiB	4.00 MiB
4096 x 4096	64 MiB	85.33 MiB	8.00 MiB	16.00 MiB
8192 x 8192	256 MiB	341.33 MiB	32.00 MiB	64.00 MiB

Assumptions: square 2D textures; RGBA8 uses 4 bytes per texel; BC1 uses 8 bytes per 4x4 block; BC5 and BC7 use 16 bytes per 4x4 block; mip figures use the conventional 4/3 approximation. Actual allocations can include alignment, arrays, cubemap faces, virtual-texture pages, or engine overhead.

Texture memory estimator

Budget a scene from the resident set outward. The correct resolution is the smallest one that preserves the required screen-space signal on target hardware.

1

Target

Choose hardware, frame budget, memory pool, and representative worst-case view.

2

Need

Estimate closest distance and screen coverage for each material family.

3

Format

Assign GPU formats by semantic, alpha need, and target support.

4

Residency

Include mips, streamability, forced resources, UI, HDRIs, and cubemap faces.

5

Measure

Profile the built content and tune the largest visible costs first.

A 4K source is not a 4K requirement. Keep high-resolution masters when useful, but export and stream according to the target view.

Production QA checklist

Source and provenance

- [ ] Untouched source archived; rights and provenance recorded.
- [ ] Physical scale, crop, projection, and UV assumptions recorded.
- [ ] White balance, exposure, lens, and perspective corrections approved.
- [ ] No unexplained baked highlights, cast shadows, or clipped channels.

Map semantics

- [ ] Each map has one declared role, range, neutral value, and sampling mode.
- [ ] Normal convention and tangent basis are documented and tested.
- [ ] Metallic classification follows the visible layer.
- [ ] Roughness or smoothness conversion occurs exactly once.

Production QA checklist

Set coherence

- [ ] Dimensions, aspect ratio, UV set, scale, and revision align.
- [ ] Features and material boundaries agree across channels.
- [ ] Seams, dilation, transparent-edge RGB, and mips pass inspection.
- [ ] Canonical masters can regenerate every packed export.

Engine import

- [ ] Shader, pipeline, engine version, and preset version are recorded.
- [ ] Color and non-color flags are correct after import.
- [ ] Packed channels and alpha mode pass a known-value probe.
- [ ] Bound material matches the canonical controlled-light preview.

Production QA checklist

Runtime and delivery

- [ ] Compression, mips, streaming, motion, and distance are approved.
- [ ] Resident memory is profiled in a representative target scene.
- [ ] Names, manifest, license, and package contents are complete.
- [ ] The delivered files were reopened and verified independently.

RELEASE RECORD

Set revision ______________________

Source hash ______________________

Export preset ______________________

Engine / shader ______________________

Reviewer ______________________

Evidence location ______________________

Troubleshooting reference

SYMPTOM

Dents look raised

LIKELY CAUSE Normal Y convention mismatch

TWO-MINUTE TEST Flip only green

PREVENTION Set the target normal preset and record it

SYMPTOM

Surface looks plastic

LIKELY CAUSE Roughness range, metallic class, or lighting

TWO-MINUTE TEST Use constant metallic 0 and sweep roughness

PREVENTION Calibrate under a fixed environment

SYMPTOM

Metal looks dark or dusty

LIKELY CAUSE Gray metallic or weak reflections

TWO-MINUTE TEST Force clean metal to 1 and load a readable HDRI

PREVENTION Classify visible layers and review with IBL

SYMPTOM

Mid-gray masks behave too dark

LIKELY CAUSE Data sampled as sRGB color

TWO-MINUTE TEST Disable sRGB for the mask

PREVENTION Declare every input as color or data

Troubleshooting reference

SYMPTOM **Roughness acts backward**

LIKELY CAUSE	Smoothness expected
TWO-MINUTE TEST	Invert once in linear data
PREVENTION	Name R versus S and use target presets

SYMPTOM **Seam appears only under light**

LIKELY CAUSE	Normal/tangent or roughness discontinuity
TWO-MINUTE TEST	Replace normal with flat and roughness with constant
PREVENTION	QA every map on final tangents

SYMPTOM **Tile repeats at distance**

LIKELY CAUSE	Memorable landmarks or macro gradient
TWO-MINUTE TEST	Preview a 4x4 grid
PREVENTION	Move macro variation to engine layers

SYMPTOM **Cutout has dark fringe**

LIKELY CAUSE	Transparent-edge RGB or alpha mismatch
TWO-MINUTE TEST	Preview over white and black
PREVENTION	Dilate edge color and match alpha mode

Troubleshooting reference

SYMPTOM	**Material sparkles in motion**
LIKELY CAUSE	High-frequency data, weak mips, or compression
TWO-MINUTE TEST	Inspect successive mips
PREVENTION	Author to texel density and keep mips

SYMPTOM	**Height barely moves mesh**
LIKELY CAUSE	Insufficient vertices or scale
TWO-MINUTE TEST	Test on a dense plane
PREVENTION	Document displacement scale and mesh need

SYMPTOM	**Channels swap after export**
LIKELY CAUSE	Wrong pack preset or binding
TWO-MINUTE TEST	Import a known RGBA probe
PREVENTION	Name channel order and version presets

SYMPTOM	**GLB loses textures**
LIKELY CAUSE	Missing URI, unsupported source, or binding
TWO-MINUTE TEST	Inspect package and material references
PREVENTION	Validate the final GLB, not only source files

Glossary

Albedo	Intrinsic diffuse reflectance term; often used loosely, so name the workflow.
Base color	Primary color input in metallic-roughness workflows; role differs for metals and dielectrics.
BRDF	Function describing reflected light for incoming and outgoing directions.
Conductor	Material with metallic optical behavior and no conventional diffuse lobe in the common model.
Dielectric	Non-metal such as stone, wood, plastic, paint, water, or glass.
Dilation	Extending valid edge colors beyond visible UV or alpha boundaries for filtering.
Energy conservation	Constraint that a material cannot reflect more light energy than it receives.
IBL	Image-based lighting from an environment texture, often used for PBR reflections.

Glossary

Linear data	Values sampled without an sRGB transfer-function decode.
Mipmap	Pre-filtered lower-resolution level used for minification and streaming.
MikkTSpace	Widely used tangent-basis method that helps normal-map portability.
Microfacet	Tiny modeled surface orientation used to explain specular reflection spread.
ORM	Pack with occlusion in R, roughness in G, and metallic in B.
Premultiplied alpha	RGB already multiplied by alpha; requires matching compositing behavior.
sRGB	Color space and nonlinear transfer function commonly used for display-oriented color textures.
Texel density	Relationship between texture resolution and physical or screen-space area.

PLAYTEX AI tools used in this book

Only tools listed in the live PLAYTEX AI /tools directory are linked here.

PBR Map Generator

Build a coherent starter stack from a prepared source.

playtex.ai/pbr-map-generator

AI Texture Generator

Create focused source variations for later calibration.

playtex.ai/ai-texture-generator

AI HDRI Generator

Create alternate environments for material stress tests.

playtex.ai/hdri-sphere-generator

PBR Engine Converter

Repack and convert canonical maps for a declared engine target.

playtex.ai/tools/pbr-engine-converter

PLAYTEX AI tools used in this book

Only tools listed in the live PLAYTEX AI /tools directory are linked here.

PLAYTEX AI Material Preflight

Inspect a set's dimensions, channels, and delivery risks.

playtex.ai/tools/material-preflight

Material Delighter / True Albedo Cleaner

Remove captured illumination before map derivation.

playtex.ai/tools/material-delighter

Alpha Bleed Fixer

Repair transparent-edge color for filtered cutouts.

playtex.ai/tools/alpha-bleed-fixer

Alpha Mode Inspector

Choose and validate cutout or blend behavior.

playtex.ai/tools/alpha-mode-inspector

PLAYTEX AI tools used in this book

Only tools listed in the live PLAYTEX AI /tools directory are linked here.

Texture Budget & VRAM Analyzer

Estimate runtime texture memory and mip overhead.

playtex.ai/tools/texture-memory-calculator

PBR Texture-Set Rotator

Rotate map stacks while preserving directional meaning.

playtex.ai/tools/pbr-texture-set-rotator

GLB Texture Extractor

Inspect and recover embedded GLB image assets.

playtex.ai/tools/glb-texture-extractor

Texture Node Editor

Build explicit, repeatable texture operations.

playtex.ai/tools/texture-node-editor

Primary sources

KHRONOS GROUP

glTF 2.0 Specification

registry.khronos.org/glTF/specs/2.0/glTF-2.0.html

KHRONOS GROUP

KTX 2.0 Specification

registry.khronos.org/KTX/specs/2.0/ktxspec.v2.html

EPIC GAMES

Physically Based Materials in Unreal Engine

dev.epicgames.com/documentation/en-us/unreal-engine/physically-based-materials-in-unreal-engine

EPIC GAMES

Using Texture Masks in Unreal Engine

dev.epicgames.com/documentation/unreal-engine/using-texture-masks-in-unreal-engine

EPIC GAMES

Texture Streaming Overview

dev.epicgames.com/documentation/unreal-engine/texture-streaming-overview-for-unreal-engine

EPIC GAMES

Texture Asset Editor

dev.epicgames.com/documentation/unreal-engine/texture-asset-editor-in-unreal-engine

Primary sources

UNITY TECHNOLOGIES

Standard Shader Material Inspector

docs.unity3d.com/Manual/StandardShaderMaterialParameters.html

UNITY TECHNOLOGIES

TextureImporter.sRGBTexture

docs.unity3d.com/ScriptReference/TextureImporter-sRGBTexture.html

UNITY TECHNOLOGIES

Normal map import settings

docs.unity3d.com/Manual/texture-type-normal-map.html

UNITY TECHNOLOGIES

URP Lit shader

docs.unity3d.com/Manual/urp/lit-shader.html

UNITY TECHNOLOGIES

HDRP Mask and Detail Maps

docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@17.0/manual/Mask-Map-and-Detail-Map.html

GODOT ENGINE

StandardMaterial3D and ORMMaterial3D

docs.godotengine.org/en/stable/tutorials/3d/standard_material_3d.html

Primary sources

GODOT ENGINE

ORMMaterial3D class reference

docs.godotengine.org/en/stable/classes/class_ormmaterial3d.html

BLENDER FOUNDATION

Principled BSDF

docs.blender.org/manual/en/dev/render/shader_nodes/shader/principled.html

BLENDER FOUNDATION

glTF 2.0 exporter material setup

docs.blender.org/manual/en/dev/addons/scene_gltf2.html

THREE.JS

MeshStandardMaterial

threejs.org/docs/pages/MeshStandardMaterial.html

THREE.JS

Color Management

threejs.org/manual/en/color-management.html

W3C

PNG Specification, Third Edition

www.w3.org/TR/png-3/

Primary sources

OPENEXR

Technical Introduction

openexr.com/en/latest/TechnicalIntroduction.html

MICROSOFT

Direct3D Block Compression

learn.microsoft.com/en-us/windows/win32/direct3d10/d3d10-graphics-programming-guide-resources-block-compression

PBRT

Roughness Using Microfacet Theory

www.pbr-book.org/4ed/Reflection_Models/Roughness_Using_Microfacet_Theory

ADOBE

Substance 3D Bitmap 2 Material parameters

helpx.adobe.com/substance-3d-b2m/desktop/parameters.html

Source review date: August 24, 2026. Renderer documentation evolves. Recheck import presets after upgrading an engine, pipeline, shader package, or web runtime.

Editorial note

This handbook was designed as a working production reference: concise enough to use beside an engine, but deep enough to explain why a map behaves the way it does.

The most durable PBR practice is not a magic value. It is a traceable pipeline: clean evidence, declared semantics, controlled translation, and repeatable verification.

The PLAYTEX AI Editorial Team reviewed the technical material against first-party specifications and current engine documentation, then separated normative rules from target conventions and editorial review windows.

THE HANDBOOK IN ONE LINE

Describe the surface. Preserve the source. Translate for the target. Verify under controlled light. Measure in the build.

Written by PLAYTEX AI Editorial Team

First edition / August 2026

BUILD MATERIALS
THAT SURVIVE THE LIGHT.

From the PBR mental model to channel packing, engine import, runtime budgets, and production QA - one calm, practical system for texture work that has to ship.

- **18 focused chapters**
- **6 material calibration cards**
- **8-target engine matrix**
- **Naming and QA sheets**

PLAYTEX AI Editorial Team

playtex.ai