Changelog

All notable changes to this project will be documented in this file.

The format is based on Keep a Changelog.

[4.0] - TBD

Added

Animation

- Revamped 3D animation storage.
  - New blend shape track to adjust blend shapes in animations more efficiently.
  - New expression-based transitions in AnimationTree state machines.
  - Support for animation compression to improve performance with long animations such as cutscenes.
  - Replaced transform tracks by position, rotation and scale tracks.
  - Removed the animation dependency on bone rests.
    - Better compatibility with models that use non-uniform scaling in animations.
    - Better compatibility with models exported from Maya and 3DS Max.
    - Easier animation reuse across different models.
    - Easier procedural generation of animations.

Core

- New TileMap and TileSet resources.
- New Vector2i, Vector3i and Rect2i types.
  - These are integer variants of Vector2, Vector3 and Rect2.
- Callable type for first-class functions (can be created with lambdas in GDScript).
- The Euler rotation order can now be adjusted in Node3D.
- New `Array.map()`, `Array.filter()` and `Array.reduce()` methods that can be used with Callables.
- Rewritten Tween with more functionality.
  - Tween is no longer a node.
  - Easier chaining of tweens.
  - Low-level tweening option to get an interpolated value directly.
  - *Existing projects will have to be modified to account for this, as automatic conversion isn't feasible.*
- New and improved IK in Skeleton2D.
  - New classes: SkeletonModifier2D, SkeletonModifierStack2D, SkeletonModification2DLookAt, SkeletonModification2DCCDIK, SkeletonModification2DFABRIK, SkeletonModification2DJiggle,

SkeletonModification2DTwoBoneIK, PhysicalBone2D, SkeletonModification2DPhysicalBones, SkeletonModification2DStackHolder.
  - New `Transform2D.looking_at()` function.
- New and improved IK in Skeleton3D.
  - New classes: SkeletonModifier3D, SkeletonModifierStack3D, SkeletonModifier3DLookAt, SkeletonModification3DCCDIK, SkeletonModification3DFABRIK, SkeletonModification3DJiggle, SkeletonModification3DTwoBoneIK, SkeletonModification3DStackHolder.
  - The Bone struct now includes a local_pose_override.
  - The Bone struct now keeps track of its children bones, if it has any.
  - Added functions to Skeleton3D for getting the forward vector using the information stored in the rest pose for the bones.
  - New `Basis.rotate_to_align()` function.
  - Refactored the BoneAttachment3D node.
  - Removed the `process_list` functions.
- New GradientTexture2D resource (useful for 2D lights, particles, ...).
- Support for gettext PO template generation from scene and script files.
  - Translation parser plugins can be written to allow extracting strings from custom file types.
- New Time singleton to replace date/time handling methods in the OS singleton.
  - Includes new methods to handle ISO 8601 timestamp conversion.
- Support for custom performance monitors.
- New `randi_range()` global scope function to return a random *integer* number within a range.
- New `randfn()` global scope function to return a floating-point number along a normal gaussian distribution.
- New `pingpong()` global scope function to return a floating-point number that increments then decrements in a "sawtooth" fashion.
- Vectors and Colors can now be clamped between two values their respective `clamp()` methods.
- New `Vector3.limit_length()` method as a Vector3 counterpart to `Vector2.limit_length()` (formerly `Vector2.clamped()`).
- New PackedArrays to replace PoolArrays.
  - 64-bit integer and float arrays are now available in addition to the existing 32-bit ones.
- New `ConfigFile.parse(data: String)` method to load a string as if it was a ConfigFile on disk.
- New `Image.save_png_to_buffer()` method to save a PNG image to memory as a PackedByteArray (instead of saving to disk).
- New File API to check, read and write symbolic links on macOS and Linux.
- Replaced GDNative with GDExtension.
  - Easier setup and compilation for various platforms.
  - Code structure is more similar to statically compiled C++ modules.
  - Lower performance overhead compared to GDNative.

Editor

- New TileMap and TileSet editors with better usability.
- Support for multiple windows.
  - Docks can be moved out of the main window into separate windows.
  - Single-window mode can be enabled in the Editor Settings to revert to the old behavior.
- Movement and scaling handles in the 2D editor (similar to the 3D editor).
- New Replace in Files dialog in the script editor to complement Find in Files.
- **macOS:** More built-in mouse cursors are now exposed (such as diagonal resize cursors).
- **macOS:** Support for building Godot with Clang sanitizers.
- **HTML5:** Support for profiling projects exported to HTML5.

Export

- **macOS:** Projects can now optionally be exported to a application bundle contained within a ZIP archive.
  - Previously, a DMG image was always used when exporting from macOS.
- **macOS:** DMG images can now be codesigned after exporting.

GDScript

- GDScript was rewritten from scratch with a cleaner approach.
  - Annotations to replace keywords in certain cases (`@export`, `@onready`, `@rpc()`, `@tool`, `@warning_ignore()`, ...).
  - Typed arrays (`var array_of_nodes: Array[Node]`). Any type can be used, including custom classes.
  - See individual progress reports for more information: #1, #2, #3.
- New documentation generation system.
  - Comments starting with `##` are considered documentation comments.
  - Documentation comments be placed before any member variable, constant, enum or function declaration, or at the top of a file.
  - Documentation comments appear in the editor help and when hovering exported properties in the inspector.

GUI

- Support for multiple windows on desktop platforms. Projects can spawn additional windows, each with their own viewport.
  - Added `NOTIFICATION_APPLICATION_FOCUS_IN` and `NOTIFICATION_APPLICATION_FOCUS_OUT` notifications for "global" project focus changes (separate from `NOTIFICATION_WM_FOCUS_IN` and `NOTIFICATION_WM_FOCUS_OUT`).
- RichTextLabel property `fit_content_height` to make the label's height fit its content automatically (not always reliable).

- RichTextLabel's `img` tag now supports an optional `color` attribute to modulate the image.
- `get_char_size()` is now exposed in Font, making it usable in DynamicFont rather than being limited to BitmapFont.
- Tree can now highlight relationship lines for the currently selected item, its parents and direct children.
  - This is used in the scene tree dock in the editor.

Import

- Support for importing lights from glTF scenes.

Input

- Support for physical (keyboard layout-independent) key codes.
  - This can be used to provide W/A/S/D controls that work on any keyboard layout.
- `DisplayServer.keyboard_get_current_layout()` and `DisplayServer.keyboard_get_layout_*()` methods to get information about keyboard layouts.
- New `Input.MOUSE_MODE_CONFINED_HIDDEN` mouse mode to combine the confined and hidden mouse modes.

Mono/C

- Support for exporting C# projects to iOS and HTML5.
- C# events can now be used to implement Godot signals.
- New Visual Studio and Visual Studio Code add-ons.

Navigation

- New NavigationServer.
  - Support for dynamic obstacle avoidance.

Networking

- Support for DTLS encryption in UDP and ENet.

Porting

- New DisplayServer abstraction, allowing for the creation of multiple windows.
  - This is used in the editor for detachable docks, but can also be used in projects.
- **Android:** Allow basic user data backup. This can be disabled in the export preset if needed.
- **Android:** Support for changing the mouse cursor shape (no custom images).

- **iOS:** The targeted device family (iPhone, iPad, iPhone and iPad) can now be specified in the export preset.

Physics

- New CharacterBody node to supersede KinematicBody.
  - Some KinematicBody features were moved to PhysicsBody.

Porting

- **Android:** Clients of the Godot library can now add their own command line arguments.

Rendering

- New Vulkan renderer.
- New OpenGL renderer, using OpenGL 3.3/OpenGL ES 3.0/WebGL 2.0 as a baseline.
  - Designed to target mobile/web platforms first, but also usable on desktop platforms.
  - Uses a low-end-friendly approach to maximize performance in simple scenes.
  - Currently supports 2D rendering only.
  - OpenGL 3D rendering is planned for a future 4.x release.
- Support for specular mapping when using 2D lighting.
- New DirectionalLight2D node for 2D lighting.
- CanvasGroup node to modulate several 2D nodes as a group (or apply shaders to them).
- Support for clipping in CanvasItem, replacing the use of Light2D as masks in a more convenient manner.
- Support for light projectors/"cookies" in OmniLight3D and SpotLight3D.
  - Only supported for lights with shadows enabled.
- 3D lights now have a Size property which can be set to simulate area lights.
  - This property also affects how fast shadow penumbras will grow over distance.
  - A shadow blur property is also available to set a constant blurring factor on a per-light basis.
- Shadow mapping with improved filtering and PCSS-like penumbra simulation.
  - Shadow normal offset bias is now implemented to avoid issues with shadow acne or peter-panning.
- New Decal node to project textures onto 3D surfaces.
- New fully real-time VoxelGI (formerly GIProbe).
  - Dynamic lights and emissive can emit GI that's updated every frame (instead of only updating sporadically).
  - Dynamic objects can receive GI and contribute to it.

- New signed distance field-based global illumination (SDFGI) for open world lighting.
  - Enabled in the WorldEnvironment. No node required, no baking.
  - Semi-realtime: dynamic objects can receive GI, but not contribute to it.
- Volumetric fog with optional GI contribution.
- Fog volumes to locally apply volumetric fog (or subtract to global fog using negative density).
- More physically accurate exponential fog to replace the old distance-based fog.
- New Aerial Scattering property in distance-based fog to fade out to the background sky instead of a fixed color.
  - Also available in volumetric fog with the Ambient Inject property.
- New GPU-based lightmapper.
  - When using a dedicated GPU, this results in much faster bake speeds compared to the CPU lightmapper.
  - Optional support for storing directional lighting information and rough reflections using spherical harmonics.
  - Improved support for lighting dynamic objects with better performance and quality.
  - In addition to automatic generation, LightmapProbe nodes can now be placed manually to provide better lighting information for dynamic objects where needed.
- Physical sky material and custom sky shaders, both supporting real-time updates.
- Global and per-instance shader uniforms.
  - This can be used to better reuse shaders, leading to improved performance.
- Support for automatically generating and using mesh LODs to improve performance.
  - Several LOD levels are generated for imported 3D scenes by default.
  - LODs are automatically used for mesh rendering using a pixel coverage-based selection algorithm.
  - Uses the meshoptimizer library.
- Support for LOD visibility ranges in GeometryInstance3D.
  - Manually authored LODs can be configured using distance and hysteresis cutoffs.
  - Can be used for HLOD setups to reduce draw calls while preserving culling opportunities when up close.
- Support for GeometryInstance3D distance fade to make distant meshes disappear smoothly without having to modify their material.
- Support for automatically generating and using shadow meshes to improve performance.
  - The generated shadow meshes are welded aggressively to improve performance with no difference in visual quality.
  - To further improve performance, hand-made shadow meshes can be

specified in the inspector in MeshInstance nodes.
- Support for rendering a viewport's 3D contents at a lower resolution to improve performance.
  - 2D elements remain at full resolution to improve perceived sharpness.
  - A scaling factor above 1.0 can be used for supersampling, which is useful to maximize quality for offline rendering.
- See individual progress reports for more information: #1, #2, #3, #4, #5, #6, #7.

Shaders

- New shader compiler rewritten from scratch.
  - Support for uniform arrays (including sampler arrays).
  - Arrays can now be passed as function parameters (including arrays of structs).
  - The return type of a function can now be an array (including arrays of structs).
  - Array size can now be optionally written before the identifier (`int[2] array;` instead of `int array[2]`).
    - This eases porting shaders from GLSL.
  - Array constructors can now be called at any time after initialization.
    - For example, `int array[3]; array = {1, 2, 3}` is now valid.
  - New `fma()` (fused multiply-add) built-in function to optimize shaders in a low-level way.
  - New built-in data (un)packing functions to optimize shaders in a low-level way.
  - Warning system for common issues such as floating-point comparison and unused variables.
  - Argument names now appear in code completion tooltips.
  - More information in the progress report.
- Add Billboard mode to visual shaders.
- The constants `PI`, `TAU` and `E` are now available in the shader language.

Miscellaneous

- The engine is now unit-tested using doctest.
  - GDScript also now has integration tests.
- Switched from Travis CI and AppVeyor to GitHub Actions.
- A Fish shell completion file is now available for the Godot editor's command line interface.

Changed

Audio

- Increased the default AudioStreamPlayer3D unit size to ($1 \rightarrow 10$) to make sounds more audible while setting up the node.

- Renamed the audio-related `FFT_Size` enum to `FFTSize` for consistency.

Core

- Tweaked the output strings to be more human-readable when printing various built-in Variant and Object types.
- Renamed File's `endian_swap` property to `big_endian` for consistency with ResourceSaver and StreamPeer.
- Renamed File's `get_len()` method to `get_length()`.
- Renamed Object's `PROPERTY_USAGE_NOEDITOR` to `PROPERTY_USAGE_NO_EDITOR`.
- Renamed `Vector2.clamped()` to `Vector2.limit_length()` to differentiate it from the new `Vector2.clamp()`.
- Renamed `rand_range()` to `randf_range()` to avoid ambiguity with the new `randi_range()` and make its return type more obvious.
- Replaced `Node.add_child_below_node()` with `Node.add_sibling()`.
- Replaced `Directory.list_dir_begin()`'s `skip_navigational` and `skip_hidden` arguments with `show_navigational` and `show_hidden`.
  - Both arguments are `false` by default, which means the default behavior is now to exclude both navigational and hidden files from the returned list.
- Renamed the built-in Quat type to Quaternion.
- Renamed the built-in Transform type to Transform3D.
- Renamed Node3D's `translation` property to `position` for consistency with Node2D.
- Moved YSort functionality to a Node2D property.
- Viewports now use a size of 512×512 by default to make them visible out of the box.
- Screen orientation is now represented as an enum in the Project Settings.
- Renamed 3D nodes to contain an explicit "3D" prefix for clarity and consistency.
- Renamed various nodes:
  - Spatial → Node3D
  - GIProbe → VoxelGI
  - BakedLightmap → LightmapGI
  - Light2D -> PointLight2D
  - VisibilityNotifier2D -> VisibleOnScreenNotifier2D
  - VisibilityNotifier3D -> VisibleOnScreenNotifier3D
  - VisibilityEnabler2D -> VisibleOnScreenEnabler2D
  - VisibilityEnabler3D -> VisibleOnScreenEnabler3D
- Renamed various resources:
  - GradientTexture -> GradientTexture1D
- Old node and resource names are automatically converted when loading scenes from Godot 3.x.

Editor

- Renewed the editor theme for a more modern design.
  - Increased icon saturation by 30% when using a dark theme.
    - Icon saturation can now be adjusted in the Editor Settings.
- Improved the audio bus editor appearance.
- Improved layout and texts of the Manage Editor Features dialog.
- Improved the Video RAM debugger usability.
  - The Video RAM tab is now refreshed automatically when switching to it.
- Hovering layer checkboxes in the inspector now results in visual feedback.
  - Clicking between two checkboxes will now enable the checkbox that was last highlighted instead of doing nothing.
- CSV profiler measures can now be saved anywhere on the filesystem, not just in the project folder.
- Improved the 2D zooming algorithm to always visit powers of two (50%, 100%, 200%, ...) and avoid floating-point precision issues.
- Times are now displayed as milliseconds in the profiler and performance monitors (instead of seconds).
- Improved the batch rename dialog usability and design consistency.
  - Clarified error messages when there are regular expression errors.
- Optimized editor icon generation to speed up editor startup.
- Script editor autocompletion now displays previews next to color constant suggestions.
- The number of replaced results now appears in place of the matches counter when replacing text in the script editor.
- Pressing Enter (or Shift + Enter) in the script editor replacement dialog now performs a forwards (or backwards) replacement operation.
- Pressing Ctrl + F now focuses the search field in the AssetLib tab.
- Pressing G now switches to the Pan mode in the 2D editor.
  - The TileMap editor's Bucket Fill shortcut was moved to B to cater for this change.
- Mouse wheel behavior for zooming in the animation behavior is now inverted.
- The Sync Scene Changes and Sync Script Changes settings' values now persist on a per-project basis instead of being always enabled by default.
- Various tooltips have been added or modified to clarify the editor operation.
- Various visual and formatting changes to the editor help to improve readability and be closer to the online class reference.
- Tweaked Camera2D editor line colors for better visibility.
- Light theme presets now use a negative contrast rate by default for a more logical preview of UI elevation.
- Increased the use of bold fonts throughout the editor.
- Revised icons for the Gradient and GradientTexture resources.
- Renamed "Identifier" to "Bundle Identifier" in the macOS and iOS export presets for clarity.
- Renamed the script editor's "Adaptive" syntax theme to "Default" and "Default" to "Godot 2", for consistency with the editor theme presets.

- Flipped the 2D editor icon to match Godot's coordinate handedness.

GUI

- Improved drive letter handling in EditorFileDialog and FileDialog.
- Container nodes (except PanelContainer) now use the Pass mouse mode by default.
- Pressing the left/right arrows while having selected text will now move the cursor to the beginning/end of the selection in LineEdit (while unselecting the text as usual).
- TextEdit's `search()` method now returns a Dictionary instead of a PackedIntArray.
- **macOS:** The Ctrl + A and Ctrl + E navigation shortcuts now work in LineEdit.

Input

- Renamed InputEventKey's `scancode` to `keycode`.
- Renamed InputMap's `get_action_list()` to `get_action_events()`.

Networking

- Optimized bandwidth usage in the high-level multiplayer API.

Physics

- Split KinematicBody into the new CharacterBody node and PhysicsBody.
- RayCast nodes are now enabled by default.
  - The `disabled` property was renamed to `enabled` with its behavior inverted.
- Renamed PlaneShape to WorldBoundaryShape.

Rendering

- Some Environment settings such as depth of field have been moved to a CameraEffects resource which is assigned to individual Camera nodes.
- The ACES Fitted tonemapping algorithm is now used in place of the old ACES algorithm.
  - The old non-fitted ACES tonemapping algorithm was removed.
- Quality settings have been moved from individual nodes and resources to the Project Settings for better centralization.
- Quality settings now have performance hints in their values' names, such as "Fast" or "Slow".

Shaders

- `DEPTH_TEXTURE` now uses normalized device coordinates between `0.0` and `1.0` (inclusive) to match Vulkan behavior.

  - This requires modifying most shaders that rely on `DEPTH_TEXTURE` to make them still work as expected.
  - Previously, coordinates would be between `-1.0` and `1.0` (inclusive) to match OpenGL behavior.
- Renamed the `.shader` file extension to `.gdshader`.
  - Existing text-based shader files will have to be renamed before loading the project in a new engine version.

Miscellaneous

- Renamed the `x11` platform to `linuxbsd` to prepare for Wayland support.
- The engine is now written in C++17.
- Python 3.6 and SCons 3.1 are now required to build Godot from source.

Removed

Buildsystem

- Removed the `server` platform in favor of disabling specific DisplayServers at build-time (e.g. `vulkan=no`).

Core

- Removed the YSort node in favor of the Node2D YSort property.
- Removed the deprecated `Color.gray()` method.
  - Use `Color.v()` for a better grayscale approximation instead.
- Removed built-in HQ2X implementation (used for crude hiDPI support in the default project theme).
  - This helps with binary size as HQ2X is made of particularly large functions.

Editor

- Removed the **Dim Dialog on Editor Popup** editor setting since it was made obsolete by the multi-window paradigm.

GUI

- Removed the ToolButton node in favor of Button.
  - Existing ToolButton nodes from Godot 3.x projects will be converted to Button nodes.

Input

- Removed the `DisplayServer.get_latin_keyboard_variant()` method (replaced by the more flexible `DisplayServer.keyboard_get_current_layout()`).

Networking

- Removed the deprecated `allow_object_decoding` property from PacketPeer.
- Removed the deprecated `sync` and `slave` high-level multiplayer keywords.

Export

- **iOS:** Remove redundant orientation export setting in favor of the orientation project setting.

Physics

- Removed the deprecated PhysicsBody `friction` and `bounce` properties (replaced by PhysicsMaterial).

Rendering

- Removed OpenGL ES 2.0 renderer (replaced by the new mobile-oriented OpenGL 3 renderer).
  - Vulkan, OpenGL 3.3, OpenGL ES 3.0 or WebGL 2.0 support is now required to run Godot.
- Removed support for 16× MSAA due to driver bugs and low performance.
  - For high-quality offline rendering, using supersampling together with 8× MSAA is a better option anyway.

Fixed

Core

- The positional command line argument now considers `.res` and `.tres` files as runnable scene formats.
  - This fixes Godot not running the main scene or a custom scene if they were saved with a `.res` or `.tres` extension.
- **macOS/Linux:** Fix the result of `Directory.get_space_left()`.
- **Windows:** Godot can now kill its own PID using `OS.kill()`.

Editor

- The Android exporter no longer reports progress on each file, greatly speeding up the exporting process.
- Searching with the Whole Words option enabled in the script editor is no longer exceedingly slow.

GUI

- Fixed OptionButton minimum size.
- TabContainer is no longer too large when tabs are hidden.

- ScrollBar now allows using `scroll_to_line()` when Scroll Active is disabled.
- DynamicFont outlines now have antialiasing disabled if it was disabled on the font itself.

Porting

- **Windows:** `OS.execute()` now only quotes command line arguments if they contain special characters.

3.2 - 2020-01-29

Added

- Support for pseudo-3D depth in 2D.
- Support for importing 3D scenes using Assimp.
  - Many formats are supported, including FBX.
- Support for generating audio procedurally and analyzing audio spectrums.
- WebRTC support.
  - Includes support for the high-level multiplayer API.
  - Supports NAT traversal using STUN or TURN.
- Support for automatically building Android templates before exporting.
  - This makes 3rd-party SDK integration easier.
- Support for texture atlases in 2D.
- Major improvements to the visual shader system. (News post 1, News post 2)
  - Redesigned visual shader editor with drag-and-drop capability.
    - Textures can be dragged from the FileSystem dock to be added as nodes.
  - Most functions available in GLSL are now exposed.
  - Many constants such as `Pi` or `Tau` can now be used directly.
  - Support for boolean uniforms and sampler inputs.
  - New Sampler port type.
  - New conditional nodes.
  - New Expression node, allowing shader code to be written in visual shaders.
  - Support for plugins (custom nodes).
    - Custom nodes can be drag-and-dropped from the FileSystem dock.
  - Ability to copy and paste nodes.
  - Ability to delete multiple nodes at once by pressing Delete.
  - The node creation menu is now displayed when dragging a connection to an empty space on the graph.
  - GLES3-only functions are now distinguished from others in the creation dialog.
  - Ability to preview the code generated by the visual shader.
  - Ability to convert visual shaders to text-based shaders.

  - See the complete list of new functions.
- Improved visual scripting.
  - Visual scripting now uses a unified graph where all functions are represented.
  - Nodes can now be edited directly in the graph.
  - Support for fuzzy searching.
  - The `tool` mode can now be enabled in visual scripts.
  - New Deconstruct node to deconstruct a complex value into a scalar value.
  - Miscellaneous UI improvements.
- Support for enabling/disabling parts of the editor or specific nodes.
  - This is helpful for education, or when working with artists to help prevent inadvertent changes.
- Language server for GDScript.
  - This can be used to get better integration with external editors.
- Version control integration in the editor.
  - This integration is VCS-agnostic (GDNative plugins provide specific VCS support).
- Improved GridMap editor.
  - The copied mesh is now displayed during pasting.
  - The duplication/paste indicator is now rotated correctly around the pivot point.
  - Ability to cancel paste and selection by pressing Escape.
  - Erasing is now done using RMB instead of Shift + RMB.
    * Freelook can still be accessed by pressing Shift + F.
- Improved MeshLibrary generation.
  - When appending to an existing MeshLibrary, previews are now only generated for newly-added or modified meshes.
  - Tweaked the previews' camera angle and light directions for better results.
  - Materials assigned to the MeshInstance instead of the Mesh are now exported to the MeshLibrary.
    * This is useful when exporting meshes from an imported scene (such as glTF), as it allows materials to persist across re-imports.
- Improved Control anchor and margin workflow.
- Network profiler.
- Improved NavigationMesh generation.
  - GridMaps can now be used to bake navigation meshes.
  - EditorNavigationMeshGenerator can now be used in `tool` scripts.
  - Support for generating navigation meshes from static colliders.
  - When using static colliders as a geometry source, a layer mask can be specified to ignore certain colliders.
  - The generator no longer relies on the global transform, making it possible to generate navmeshes on nodes that are not in the scene tree.
  - Navigation gizmos are now updated after every new bake.

- Support for skinning in 3D skeletons.
- CameraServer singleton to retrieve images from mobile cameras or webcams as textures.
- A crosshair is now displayed when using freelook in the 3D editor.
- Project camera override button at the top of the 2D and 3D editors.
  - When enabled, the editor viewport's camera will be replicated in the running project.
- RichTextLabel can now be extended with real-time effects and custom BBCodes.
  - Effects are implemented using the ItemFX resource.
- `[img=<width>x<height>]` tag to resize an image displayed in a RichTextLabel.
  - If `<width>` or `<height>` is 0, the image will be adjusted to keep its original aspect.
- Revamped node connection dialog for improved ease of use.
- The Signals dock now displays a signal's description in a tooltip when hovering it.
- Input actions can now be reordered by dragging them.
- Animation frames can now be reordered by dragging them.
- Ruler tool to measure distances and angles in the 2D editor.
- "Clear Guides" menu option in the 2D editor to remove all guides.
- The 2D editor grid now displays a "primary" line every 8 lines for easier measurements.
  - This value can be adjusted in the Configure Snap dialog.
- Projects can now have a description set in the Project Settings.
  - This description is displayed as a tooltip when hovering the project in the Project Manager.
- All Variant types can now be added as project settings using the editor (instead of just `bool`, `int`, `float` and `String`).
- Pressing Ctrl + F now focuses the search field in the Project Settings and Editor Settings.
- Quick Open dialog (Shift + Alt + O) to open any resource in the project.
  - Unlike the existing dialogs, it's not limited to scenes or scripts.
- Ability to convert a Sprite to a Mesh2D, Polygon2D, CollisionPolygon2D or LightOccluder2D.
- MultiMeshInstance2D node for using MultiMesh in 2D.
- PointMesh primitive.
  - Drawn as a rectangle with a constant size on screen, which is cheaper compared to using triangle-based billboards.
- 2D polygon boolean operations and Delaunay triangulation are now available in the Geometry singleton.
- New convex decomposition using the V-HACD library.
  - Can decompose meshes into multiple convex shapes for increased accuracy.
- Support for grouping nodes in the 3D editor.
- "Slow" modifier in freelook (accessed by holding Alt).

- The 2D editor panning limits can now be disabled in the Editor Settings.
- "Undo Close Tab" option in the scene tabs context menu.
- The editor is now capped to 20 FPS when the window is unfocused.
  - This decreases CPU/GPU usage if something causes the editor to redraw continuously (such as particles).
- The editor's FPS cap can now be adjusted in the Editor Settings (both when focused and unfocused).
- Version information is now displayed at the bottom of the editor.
  - This is intended to make the Godot version easily visible in video tutorials.
- Support for constants in the shader language.
- Support for local and varying arrays in the shader language.
- Support for `switch` statements in the shader language.
- Support for `do {...} while (...)` loops in the shader language.
  - Unlike `while`, the expression in the `do` block will always be run at least once.
- Support for hexadecimal number literals in the shader language.
- Ported several GLES3 shader functions such as `round()` to GLES2.
- `SHADOW_VEC` shader parameter to alter 2D shadow computations in custom shaders.
- Filter search box in the remote scene tree dock.
- Ability to expand/collapse nodes recursively in the scene tree dock by holding Shift and clicking on a folding arrow.
- Support for depth of field, glow and BCS in the GLES2 renderer.
- MSAA support in the GLES2 renderer.
- Ability to render viewports directly to the screen in the GLES2 renderer.
  - This can be faster on low-end devices, but it comes at a convenience cost.
- Project settings to set the maximum number of lights and reflections in the GLES3 renderer.
  - Decreasing these values can lead to faster shader compilations, resulting in lower loading times.
- Heightmap collision shape for efficient terrain collisions.
- AStar2D class, making A* use easier in 2D.
- Disabled collision shapes can now be added directly, without having to disable them manually after one step.
- Context menu options to close other scene tabs, scene tabs to the right, or all scene tabs.
- The audio bus volumes can now be snapped by holding Ctrl while dragging the slider.
- Hovering an audio bus' volume slider now displays its volume in a tooltip.
- Values in the Gradient and Curve editors can now be snapped by holding Ctrl.
  - Precise snapping can be obtained by holding Shift as well.
- Support for snapping when scaling nodes in the 2D editor.
- Precise snapping in the 3D editor when holding Shift.

- "Align Rotation with View" in the 3D editor.
  - Unlike "Align Transform with View", only the selected node's rotation will be modified.
  - "Align Selection with View" has been renamed to "Align Transform with View".
- All 3D gizmos now make use of snapping if enabled.
- CSG shapes are now highlighted with a translucent overlay when selected.
  - Shapes in Union mode will use a blue overlay color by default.
  - Shapes in Subtraction mode will use an orange overlay color by default.
  - Shapes in Intersection mode will use a white overlay color.
- Ability to move a vertex along a single axis when holding Shift in polygon editors.
- Support for binary literals in GDScript (e.g. `0b101010` for `42`).
- AutoLoads can now be used as a type in GDScript.
- Ability to define script templates on a per-project basis.
  - Template files should be placed into a `script_templates/` directory in the project and have an extension that matches the language (`.gd` for GDScript, `.cs` for C#).
  - The path to the script templates directory can be changed in the Project Settings.
- Ability to limit the minimum and maximum window size using `OS.set_min_window_size()` and `OS.set_max_window_size()`.
- `Node.process_priority` property to set or get a node's processing priority.
  - This was previously only available as `Node.set_process_priority()` (without an associated getter).
- `Node.editor_description` property for documentation purposes.
  - When hovering a node with a description in the scene tree dock, the description will be displayed in a tooltip.
- `Button.keep_pressed_outside` property to keep a button pressed when moving the pointer outside while pressed.
- `Button.expand_icon` property to make a button's icon expand/shrink with the button's size.
- `Popup.set_as_minsize()` method to shrink a popup to its minimum size.
- `Tree.get_icon_modulate()` and `Tree.set_icon_modulate()` methods to change an icon's color in a Tree.
- `Tree.call_recursive()` method to call a method on a TreeItem and its children recursively.
- `Light.use_gi_probe` property to exclude specific lights from GIProbe computations.
- TranslationServer method `get_loaded_locales()` to retrieve the list of languages with a translation loaded.
- `FRUSTUM` 3D camera mode to create tilted frustums for mirror or portal effects.
- `CanvasItem.draw_rect()` now has `width` and `antialiased` properties to match `draw_line()`'s functionality.
- `Engine.get_idle_frames()` and `Engine.get_physics_frames()` to get

the number of idle and physics frame iterations since the project started.
    - Unlike `Engine.get_frames_drawn()`, `Engine.get_idle_frames()` will be incremented even if the render loop is disabled.
- `Engine.get_physics_interpolation_fraction()` to get the fraction through the current physics tick at the time of the current frame.
    - This can be used to implement fixed timestep interpolation.
- Support for shadow-to-opacity in 3D to render shadows in augmented reality contexts.
- Ability to change a Position2D gizmo's size.
- New Vector2 and Vector3 methods:
    - `move_toward()` to retrieve a vector moved towards another by a specified number of units.
    - `direction_to()` to retrieve a normalized vector pointing from a vector to another.
        * This is a shorter alternative to `(b - a).normalized()`.
- AStar functions `set_point_disabled()` and `is_point_disabled()` to selectively disable points.
- Tween now emits a `tween_all_completed` signal when all tweens are completed.
- `Input.get_current_cursor_shape()` to retrieve the current cursor shape.
- `InputEventAction` now has a `strength` property to simulate analog inputs.
- `String.repeat()` method to repeat a string several times and return it.
- `String.count()` method to count the number of occurrences of a substring in a string.
- `String.humanize_size()` method to display a file size as an human-readable string.
- `String.strip_escapes()` to strip non-printable escape characters from a string, including tabulations and newlines (but not spaces).
- `String.sha1_text()` and `String.sha1_buffer()` methods to return a string's SHA-1 hash.
- Line2D `clear_points()` method to clear all points.
- Line2D now has a "Width Curve" property to make its width vary at different points.
- `assert()` now accepts an optional second parameter to display a custom message when the assertion fails.
- `posmod()` built-in GDScript function that behaves like `fposmod()`, but returns an integer value.
- `smoothstep()` built-in GDScript function for smooth easing of values.
- `lerp_angle()` built-in GDScript function to interpolate between two angles.
- `ord()` built-in GDScript function to return the Unicode code point of an 1-character string.
- `PoolByteArray.hex_encode()` method to get a string of hexadecimal numbers.
- `Font.get_wordwrap_string_size()` method to return the rectangle size

needed to draw a word-wrapped text.
- `Camera.get_camera_rid()` method to retrieve a Camera's RID.
- `Array.slice()` method to duplicate a subset of an Array and return it.
- The GraphEdit box selection colors can now be changed by tweaking the `selection_fill` and `selection_stroke` theme items.
- Toggleable HSV mode for ColorPicker.
- ColorPicker properties to toggle the visibility and editability of presets.
- The default ColorPicker mode (RGB, HSV, RAW) can now be changed in the Editor Settings.
- ColorPicker now displays an indicator to denote "overbright" colors (which can't be displayed as-is in the preview).
- Hovering a Color property in the editor inspector now displays a tooltip with the exact values.
- `Color.transparent` constant (equivalent to `Color(1, 1, 1, 0)`).
- `KinematicBody.get_floor_normal()` and `KinematicBody2D.get_floor_normal()` to retrieve the collided floor's normal.
- `VehicleWheel.get_rpm()` method to retrieve a vehicle wheel's rotations per minute.
- Per-wheel throttle, brake and steering in VehicleBody.
- `GeometryInstance.set_custom_aabb()` to set a custom bounding box (used for view frustum culling).
- `FuncRef.call_funcv()` to call a FuncRef with an array containing arguments.
  - In contrast to `FuncRef.call_func()`, only a single array argument is expected.
- `Mesh.get_aabb()` is now exposed to scripting.
- `PhysicalBone.apply_impulse()` and `PhysicalBone.apply_central_impulse()` methods to push ragdolls around.
- `ProjectSettings.load_resource_pack()` now features an optional `replace_files` argument (defaulting to `true`), which controls whether the loaded resource pack can override existing files in the virtual filesystem.
- `SpinBox.apply()` method to evaluate and apply the expression in the SpinBox's value immediately.
- `ConfigFile.erase_section_key()` method to remove a single key from a ConfigFile.
- `OS.execute()` now returns the process' exit code when blocking mode is enabled.
- `OS.is_window_focused()` method that returns `true` if the window is currently focused.
  - Tracking the focus state manually using `NOTIFICATION_WM_FOCUS_IN` and `NOTIFICATION_WM_FOCUS_OUT` is no longer needed to achieve this.
- `OS.low_processor_mode_sleep_usec` is now exposed as a property.
  - This makes it possible to change its value at runtime, rather than just defining it once in the Project Settings.
- `SceneTree.quit()` now accepts an optional argument with an exit code.
  - If set to a value greater than or equal to 0, it will override the

`OS.exit_code` property.

- `VisualServer.get_video_adapter_name()` and `VisualServer.get_video_adapter_vendor()` methods to retrieve the user's graphics card model and vendor.
- `VisualServer.multimesh_create()` is now exposed to scripting.
- Ability to override how scripted objects are converted to strings by defining a `_to_string()` method.
- Export hints for 2D and 3D physics/render layers.
- Editor plugins can now add new tabs to the Project Settings.
- Standalone ternary expression warning in GDScript.
- Variable shadowing warning in GDScript.
  - Will be displayed if:
    - a block variable shadows a member variable,
    - a subclass variable shadows a member variable,
    - a function argument shadows a member variable.
- Script reflection methods are now exposed to GDScript.
  - See `Script.get_script_property_list()`, `Script.get_script_method_list()`, `Script.get_script_signal_list()`, `Script.get_script_constant_map()` and `Script.get_property_default_value()`.
- `randfn(mean, deviation)` method to generate random numbers following a normal Gaussian distribution.
- Ability to read the standard error stream when using `OS.execute()` (disabled by default).
- Option to disable boot splash filtering (nearest-neighbor interpolation).
- The GridMap editor now offers a search field and size slider.
- DynamicFont resources now have a thumbnail in the editor.
- Minimap in the script editor.
- Bookmarks in the script editor for easier code navigation.
- Filter search box for the script list and member list.
- Singletons and `class_name`-declared classes are now highlighted with a separate color in the script editor.
- The editor help now displays class properties' default and overridden values.
- The script editor's Find in Files dialog can now search in user-defined file types (`editor/search_in_file_extensions` in the Project Settings).
- The script editor search now displays the number of matches.
- The script editor search now selects the current match for easier replacing.
- "Evaluate Expression" contextual option in the script editor.
  - This option evaluates the selected expression and replaces it (e.g. `2 + 2` becomes `4`).
- Autocompletion support for `change_scene()`.
- Ability to skip breakpoints while debugging.
- Drag-and-drop support in the TileSet editor.
- Ability to attach scripts to nodes by dragging a name from the script list to a node in the scene tree.
- Icons are now displayed next to code completion items, making their type easier to distinguish.
- TileMap property `centered_textures` can be used to center textures on

their tile, instead of using the tile's top-left corner as position for the texture.
- "Ignore" flag to ignore specific tiles when autotiling in the TileMap editor.
- Keyboard shortcuts to rotate tiles in the TileMap editor.
  - Default shortcuts are A (rotate left), S (rotate right), X (flip horizontally), Y (flip vertically).
- Ability to keep a node's local transform when reparenting it by holding Shift.
- Basis constants `IDENTITY`, `FLIP_X`, `FLIP_Y`, `FLIP_Z`.
- Ability to create sprite frames in AnimatedSprite from a sprite sheet.
- `frame_coords` property in Sprite and Sprite3D to set/get the coordinates of the frame to display from the sprite sheet.
- `billboard` property in Sprite3D.
- Reimplemented support for editing multiple keys at once in the animation editor.
- Support for FPS snapping in the Animation editor.
- Autokeying in the Animation editor.
  - Keyframes will be created automatically when translating, rotating or scaling nodes if a track exists already.
  - Keys must be inserted manually for the first time.
- AnimationNodeBlendTreeEditor improvements.
  - Ability to exclude multiple selected nodes at once.
  - Context menu to add new nodes (activated by right-clicking).
- The AnimationPlayer Call Method mode is now configurable.
  - Method calls can be "deferred" or "immediate", "deferred" being the default.
- OccluderPolygon2D is now draggable in the editor.
- The tooltip position offset is now configurable.
- The default cursor used when hovering RichTextLabels can now be changed.
- "Dialog Autowrap" property in AcceptDialog to wrap the label's text automatically.
- The 2D editor's panning shortcut can now be changed.
- The shortcuts to quit the editor can now be changed.
- Support for emission masks in CPUParticles2D.
- `direction` property in CPUParticles and ParticlesMaterial.
- `lifetime_randomness` property in CPUParticles and ParticlesMaterial.
- CPUParticles now uses a different gizmo icon to distinguish them from Particles.
- "Restart" button to restart particle emission in the editor.
- AnimatedSprites' animations can now be played backwards.
- TextureRects can now have their texture flipped horizontally or vertically.
- StyleBoxFlat shadows can now have an offset.
- StyleBoxFlat now computes UV coordinates for its `canvas_item` vertices, which can be used in custom shaders.
- Profiler data can now be exported to a CSV file.
- The 2D polygon editor now displays vertex numbers when hovering vertices.

- RectangleShapes now have a third handle to drag both axes at once.
- Global class resources are now displayed in the Resource property inspector.
- Double-clicking an easing property in the inspector will now make the editor display a numeric field.
  - This makes it easier to enter precise values for properties such as light attenuation.
- `interface/editor/default_float_step` editor setting to configure floating-point values' default step in the Inspector.
- Audio buses are now stylized to look like boxes that can be dragged.
- The default audio bus layout file path can now be changed in the Project Settings.
- The LineEdit and TextEdit controls now display their contextual menu when pressing the Menu key.
- `shortcut_keys_enabled` and `selecting_enabled` LineEdit and TextEdit properties to disable keyboard shortcuts and selecting text.
- The LineEdit "disabled" font color can now be changed.
- The TextEdit "readonly" font color can now be changed.
- LineEdit can now have its `right_icon` set in scripts.
- The `nine_patch_stretch` TextureProgress property now enables stretching when using a radial fill mode.
- Support for loading and saving encrypted files in ConfigFile.
- `get_path()` and `get_path_absolute()` are now implemented in FileAccessEncrypted.
- "Disabled" attenuation model for AudioStreamPlayer3D, making the sound not fade with distance while keeping it positional.
- AudioEffectPitchShift's FFT size and oversampling are now adjustable.
- TextEdit's tab drawing and folding is now exposed to GDScript.
- Orphan node monitor in the Performance singleton.
  - Counts the number of nodes that were created but aren't instanced in the scene tree.
- Ability to change eye height in VR.
- CSV files can now be imported as non-translation files.
- Scene resources such as materials can now be imported as `.tres` files.
- Support for importing 1-bit, 4-bit and 8-bit BMP files.
  - Size dimensions must be a multiple of 8 for 1-bit images and 2 for 4-bit images.
- `use_lld=yes` flag to link with LLD on Linux when compiling with Clang.
  - This results in faster iteration times when developing Godot itself or modules.
- `use_thinlto=yes` flag to link with ThinLTO when using Clang.
- Multicast support in PacketPeerUDP.
- `NetworkedMultiplayerEnet.server_relay` property to disable server relaying.
  - This can be used to increase security when building a fully-authoritative server.
- Automatic timeout for TCP connections (defaults to 30 seconds, can be

changed in the Project Settings).
- `HTTPRequest.timeout` property (defaults to 0, which is disabled).
- `HTTPRequest.download_chunk_size` property.
  - This value can be adjusted to reduce the allocation overhead and file writes when downloading large files.
  - The default value was increased for faster downloads (4 KB → 64 KB).
- WebSocket improvements.
  - Support for SSL in WebSocketServer.
  - WebSocketClient can now use custom SSL certificates (except on HTML5).
  - WebSocketClient can now define custom headers.
- The editor now features a built-in Web server for testing HTML5 projects.
- Button to remove all missing projects in the Project Manager.
- Reimplemented support for embedding project data in the PCK file.
- Ability to take editor screenshots by pressing Ctrl + F12.
- Editor plugins can now set the current active editor as well as toggle the distraction-free mode.
- **Android:** Support for adaptive icons.
  - All icon densities are now generated automatically by the exporter.
  - Only 3 images now need to be supplied to support all icon formats and densities (legacy icon, adaptive foreground, adaptive background).
- **Android:** Support for the Oculus Mobile SDK.
- **Android:** Support for requesting permissions at runtime.
- **Android:** `NOTIFICATION_APP_PAUSED` and `NOTIFICATION_APP_RESUMED` notifications are now emitted when the app is paused and resumed.
- **Android:** Support for pen input devices.
- **Android/iOS:** Support for vibrating the device.
- **HTML5:** Partial clipboard support.
- **iOS:** Support for ARKit.
- **iOS:** `OS.get_model_name()` now returns a value with the device name.
- **iOS:** The Home indicator is now hidden by default to avoid being in the way of the running project.
  - It can be restored in the Project Settings.
- **Windows:** Ability to toggle the console window in the Editor Settings.
- **Windows:** Project setting to enable Vsync using the compositor (DWM), disabled by default.
  - On some hardware, this may fix stuttering issues when running a project in windowed mode.
- **Windows:** Support for code signing using `signtool` on Windows and `osslsigncode` on other platforms.
- **Windows:** Support for using Clang and ThinLTO when compiling using MinGW.
- **Windows/macOS:** `OS.set_native_icon()` method to set an `.ico` or `.icns` window/taskbar icon at runtime.
- **Windows/macOS/X11:** Support for graphic tablet pen pressure and

tilt in InputEventMouseMotion.
- **macOS:** LineEdit now supports keyboard shortcuts commonly available on macOS.
- **macOS:** Multiple instances of the editor can now be opened at once.
- **macOS:** Recent and favorite projects are now listed in the project manager dock menu.
- **macOS:** The list of open scenes is now displayed in the editor dock menu.
- **macOS:** Support for modifying global and dock menus.
- **macOS:** Improved support for code signing when exporting projects.
- **macOS:** Support for defining camera and microphone usage descriptions when exporting a project.
- **macOS/X11:** A zsh completion file for the editor is now available.
- **X11:** The instance PID is now set as the `_NET_WM_PID` window attribute, so that external programs can easily access it.
- **Mono:** Support for exporting to Android and HTML5.
- **Mono:** Support for using Rider as an external editor.
- **Mono:** Support for attaching external profilers like dotTrace using the `MONO_ENV_OPTIONS` environment variable.
- **Mono:** New DynamicGodotObject class to access dynamic properties from scripts written in GDScript.
- **Mono:** Support for resource type hints in exported arrays.
- **Mono:** New `mono/unhandled_exception_policy` project setting to keep running after an unhandled exception.
- **Mono:** New Godot constants to conditionally react to system variables at compile-time.
- **Mono:** Support for Visual Studio 2019's MSBuild.

Changed

- Tween and Timer now display an error message if they are started without being added to the scene tree first.
- Tweaked Timer's wait time property hint to allow values with 3 decimals and above 4096.
- Functions called from a signal can no longer disconnect the node from the signal they're connected to (unless using `call_deferred()`).
- Tabs and space indentation can no longer be mixed in the same GDScript file.
  - Each file must now use only tabs or spaces for indentation (not both).
- `assert()` in GDScript must now always be used with parentheses.
  - `assert(true)` is still valid, but `assert true` isn't valid anymore.
  - This is to account for the optional second parameter that defines a custom message.
- The "Trim" and "Normalize" WAV import options are now disabled by default.
  - This makes the default behavior more consistent with Ogg import.
- Ogg samples now have an icon in the editor, like WAV samples.

- Camera2D drag margins are now disabled by default.
  - If porting a project from Godot 3.1 where drag margins were used, these must be enabled manually again.
- The Camera2D Offset property now ignores the Limit property.
  - To get the old behavior back, move the camera itself instead of changing the offset.
- `Camera.project_position()` now requires a second `depth` argument to determine the distance of the point from the camera.
  - To get the old behavior back, pass the Camera's `near` property value as the second argument.
- `Skeleton.set_bone_global_pose()` was replaced by `Skeleton.set_bone_global_pose_override()`.
- UDP broadcasting is now disabled by default and must be enabled by calling `set_broadcast_enabled(true)` on the PacketPeerUDP instance.
- The editor and project manager now open slightly faster.
- Improved the Project Manager user interface.
  - New, simpler design with more space available for the project list.
  - Improved reporting of missing projects.
  - The search field is now focused when starting the Project Manager if there is at least one project in the list.
  - The search field now searches in both the project name and path.
    - If the search term contains a `/`, the whole path will be used to match the search them. Otherwise, only the last path component will be searched in.
- Refactored the Project Manager to be more efficient, especially with large project lists.
- Images in the Project Manager and Asset Library are now resized with Lanczos filtering for a smoother appearance.
- The editor now uses the font hinting algorithm that best matches the OS' default.
  - Hinting is set to "None" on macOS, and set to "Light" on Windows and Linux.
  - This can be changed in the Editor Settings.
- The editor window dimming when a popup appears is now less intense (60% → 50%).
  - The animation was also removed as it made the editor feel sluggish at lower FPS.
- Several editor menus have been reorganized for consistency and conciseness.
- Undo/Redo now supports more actions throughout the editor.
- Increased the height of the ItemList editor popup.
  - This makes it easier to edit large amounts of items.
- Opening a folder in FileDialog will now scroll back to the top.
- Folder icons in FileDialog can now be displayed with a different color using the `folder_icon_modulate` constant, making them easier to distinguish from files.
  - Folder icons in editor file dialogs are now tinted with the accent color.
- Improved colors in the light editor theme for better readability and consis-

tency.
- Improved A* performance significantly by using a binary heap and OA-HashMap.
- Tweaked the AABB transform algorithm to be ~1.2 times faster.
- Optimized the variant reference function, making complex scripts slightly faster.
- Disabled high-quality voxel cone tracing by default.
  - This makes GIProbe much faster out of the box, at the cost of less realistic reflections.
- Lowered the default maximum directional shadow distance (200 → 100).
  - This makes directional shadow rendering consistent between the editor and running project when using the default Camera node settings.
- Tweaked the default depth fog maximum distance to be independent of the Camera's `far` value (0..100).
  - This makes fog display consistent between the editor and a running project.
- Tweaked the default height fog values to be more logical (0..100 → 10..0).
  - This means height fog will be drawn from top-to-bottom, instead of being drawn from bottom-to-top.
- Significantly improved SSAO performance by using a lower sample count.
  - SSAO now uses 3×3 blurring by default, resulting in less visible noise patterns.
- When "Keep 3D Linear" is enabled, colors are no longer clamped to [0, 1] when using Linear tonemapping.
  - This allows rendering HDR values in floating-point texture targets for further processing or saving HDR data into files.
- The lightmap baker now calculates lightmap sizes dynamically based on surface area.
- Improved 3D KinematicBody performance and reliability.
- Orbiting in the 3D editor can now be done while holding Alt, for better compatibility with graphics tablets.
- Keys and actions are now released when the window loses focus.
- Tweens can now have a duration of 0.
- Particles and CPUParticles' Sphere emission shape now uses an uniform density sphere.
- `Viewport.size_override_stretch` is now exposed as a property (rather than just setter/getter methods).
- One-click deploy to Android now requires just one click if only one device is connected.
- The Project Manager will now infer a project name from the project path if the name was left to the default value.
- The WebSockets implementation now uses the smaller wslay library instead of libwebsockets.
- Box selections in the editor now use a subtle outline for better visibility.
- Most 2D lines are now antialiased in the editor.
- CheckButtons now use a simpler design in the editor.

- Messages originating from the editor are now faded in the editor log.
  - This makes messages printed by the project stand out more.
- Folding arrows in the editor inspector are now displayed at the left for consistency with other foldable elements.
- Hovering or dragging guides in the 2D editor will now turn the cursor into a "resizing" shape.
- The editor update spinner is now hidden by default.
  - It can be enabled again in the Editor Settings.
- The "Update Always" option is now editor-wide instead of being project-specific.
- ColorPicker, OptionButton and MenuButton now use toggle mode, making them appear pressed when clicked.
- The ColorPicker preview was moved below the picker area to be closer to the sliders.
- Increased the Light2D height range from -100..100 to -2048..2048.
  - Lower and higher values can be entered manually too.
- Decreased the `rotation_degrees` range in various nodes to -360..360 to be easier to adjust using the slider.
  - Lower and higher values can still be entered manually, which is useful for animation purposes.
- The default RichTextLabel color is now `#ffffff`, matching the default Label color for better consistency.
- SpinBoxes now calculate the entered value using the Expression class.
  - For example, writing `2 + 2` in a SpinBox then pressing Enter will result in `4`.
- Saved resources no longer contain dependency indices and metadata such as node folding, resulting in more VCS-friendly files.
- The script editor's line length guideline is now enabled by default.
- The script editor state (such as breakpoints or the current line) is now preserved across editor sessions.
- The script editor's "Auto Brace Complete" setting is now enabled by default.
- The scripts panel toggle button is now located at the bottom-left of the script editor (instead of the File menu).
- Editor plugins can now be enabled without having an init script defined.
- Custom nodes added by plugins now have a translucent script icon in the scene tree dock.
- `EditorInterface.get_current_path()` to get the full path currently displayed in the FileSystem dock in an editor plugin.
- Copy constructors are now allowed for built-in types in GDScript.
  - This allows constructs such as `Vector2(Vector2(12, 34))`, which may be useful to simplify code in some cases.
- `weakref(null)` is now allowed in GDScript.
  - This makes checking for a valid reference more concise, as `if my_ref.get_ref()` is now sufficient (no need for `if my_ref and my_ref.get_ref()`).

- The number of signal connections and groups is now displayed in a tooltip when hovering the associated buttons in the scene tree dock.
- The right mouse button can now be used to pan in the 2D editor.
  - This is to improve usability when using a touchpad.
  - The middle mouse button can still be used to pan in the 2D editor.
- Zooming is now allowed while panning in the 2D editor.
- When the "Scroll To Pan" editor setting is enabled, the 2D editor can now be zoomed in by holding Ctrl and scrolling the mouse wheel.
- Zoom percentages in the 2D editor are now relative to the editor scale if the editor scale is higher than 100%.
- The 2D editor now displays the current zoom percentage.
  - The zoom percentage can be clicked to reset the zoom level to 100%.
- Improved sorting options in the Asset Library.
- Images now load faster in the Asset Library.
- A loading placeholder is now displayed while icons are loading in the Asset Library.
- Images failing to load in the Asset Library display a "broken file" icon.
- Improved the Asset Library page loading transitions.
- Tweaked the Asset Library detail page layout for better readability.
- Audio mixer faders now use a non-linear algorithm to better fit human hearing.
- Tooltips now appear faster when hovering elements in the editor (0.7 seconds → 0.5 seconds).
- Increased the low-processor usage mode's default maximum refresh rate (125 FPS → 144 FPS).
  - This makes the editor feel slightly smoother on 144 Hz displays.
- Tree scrolling when dragging now uses a larger drag margin, making drag-and-drop more convenient.
- Holding Ctrl now toggles snapping in GraphEdit.
- Improved the timeline's appearance in the animation editor.
- Improved snapping in the animation editor.
  - Snapping can be toggled temporarily by holding the Ctrl key.
  - Snapping can be made more precise by holding the Shift key.
  - Timeline snapping is now toggled by the Snap setting (like when moving keyframes).
- Keyframes are now easier to select in the animation editor.
- Selected keyframes now appear slightly larger in the animation editor.
- Boolean and color keyframe icons are now aligned to other keyframes in the animation editor.
- The Animation editor's line widths are now resized to match the editor scale.
- BPTC compression is now available for all HDR image formats.
- `Image.save_exr()` to save an image in EXR format, which supports high bit depths.
- Improved path and polygon editors.
  - New handle icons for path and polygon points.

  - Smooth path point and curve tangents now use different icons to be distinguished from sharp points.
  - Tangent lines are now gray in the Path2D and Path editors.
  - Path2D lines are now antialiased.
- Increased the TileSet and polygon UV editor's maximum zoom levels (400% → 1600%).
- Decreased the maximum allowed StyleBoxFlat corner detail (128 → 20).
  - This prevents slowness and glitches caused by using overly detailed corners.
- 3D collision shapes and RayCasts are now drawn in gray when disabled.
- Improved RayCast2D and one-way collision drawing.
  - Disabled RayCast2Ds are now displayed in gray.
  - One-way collision arrows are now orange by default, making them easier to distinguish them from RayCast2Ds.
  - Tweaked RayCast2D and one-way collision line shapes to look more like arrows.
- Improved rendering in the curve editor.
  - The grid is now rendered correctly when using a light theme.
  - The main line and edge line colors have been swapped for better visibility.
  - Tangent line widths are now resized to match the editor scale.
- Improved rendering in the performance monitor.
  - Dark colors are now used on light backgrounds for better visibility.
  - Graph lines are now thinner and opaque.
  - Graph line widths are now resized to match the editor scale.
  - Rounded values now display trailing zeroes to make their precision clearer.
- TileMap support for transform operations on cell textures bigger than the cell size has been reworked to properly support isometric tiles.
  - Breaks compatibility with some TileMaps from previous Godot versions. An opt-in `compatibility_mode` property can be used to restore the previous behavior.
- Some TileMap editor options were moved to the toolbar.
- The TileMap editor now displays coordinate information in the 2D viewport's bottom-left corner.
  - This fixes the TileMap editor width changing when hovering tiles in a small window.
- Brackets are now only inserted when necessary when autocompleting methods in the script editor.
- Improved dialogs when saving or removing an editor layout.
- Whitespace-only selections no longer cause the script editor to highlight all occurrences.
- Saving a script will now add a newline at the end of file if none was present already.
- Reorganized sections in the editor help to be in a more logical order.
- The editor help now uses horizontal margins if the screen is wide enough.

  - This makes sure lines keep a reasonable length for better readability.
- Increased line spacing in the editor help and asset library descriptions.
- The editor help now displays bold text using a bold font (instead of using a monospace font).
- The editor help now displays code using a slightly different color to be easier to distinguish.
- The editor help now displays types after parameter names to follow the GDScript static typing syntax.
- Editor help is now accessed using Shift + F1, for consistency with other applications.
  - Contextural help is now accessed using Alt + F1 to accommodate for this change.
- The script editor's Find in Files dialog is now always available, even when no script is opened.
- Pressing Shift + Enter in the script editor Find dialog will now go to the previous match.
- Improved the node deletion confirmation message.
  - If there is only one node to delete, its name is displayed in the message.
  - If there is more than one node to delete, the number of nodes to delete is displayed.
- Improved the "Snap Object to Floor" functionality in the 3D editor.
  - An error message is now displayed if no nodes could be snapped.
  - Increased the maximum snapping height (10 → 20).
  - Increased the maximum snapping tolerance (0.1 → 0.2).
- 2D/3D selections, rotations and selected texts are now highlighted with the editor theme's accent color.
- 3D light gizmos are now tinted using the light's color, making navigation easier while using the unshaded display mode.
- Improved the 3D light and AudioStreamPlayer3D gizmos to better represent their depth in the 3D world.
- Tweaked the 3D manipulator gizmo's colors for better visibility.
- Tweaked the 2D and 3D axis colors for consistency with gizmo colors.
- Increased the default 3D manipulator gizmo opacity (0.2 → 0.4).
- The multiline text editor popup dialog's width is now capped on large displays.
  - This prevents lines from becoming very long, which could hamper text readability.
- Non-printable escape characters are now stripped when pasting text into a LineEdit.
- The TextEdit caret color now matches the default font color, making it easier to see.
- Empty exported NodePath properties now return `null` instead of `self`.
- Built-in scripts are no longer allowed to use `class_name` as it wasn't working properly.
- The second parameter of `substr()` is now optional and defaults to `-1`.
- More editor actions can now have shortcuts assigned (such as Revert Scene

or Export).
- The project export path may now be written in a relative path.
  - Directories will be created recursively if the target directory doesn't exist.
- Items in the FileSystem dock can now be deselected by clicking empty space.
- "Set as Main Scene" context option for scenes in the FileSystem dock.
- The unused class variable GDScript warning is now disabled by default due to false positives.
- Warning-ignore comments now allow whitespace after the `#` character.
- Improved error reporting in the Particles emission point creation dialog.
- The number of warnings and errors that can be received in the remote debugger is now capped per second rather than per frame.
  - The default limit is 100 errors and 100 warnings per second, making it possible for the script editor to report up to 100 warnings before having messages hidden.
- UTF-8 characters are now supported in input action names.
- All platforms now use the `custom_template` property in each export preset to store the path to the custom export template (instead of `custom_package` for some platforms).
- Tween methods' `trans_type` and `ease_type` arguments are now optional, defaulting to `TRANS_LINEAR` and `EASE_IN_OUT` respectively.
- `PCKPacker.pck_start()` and `PCKPacker.flush()`'s `alignment` and `verbose` arguments (respectively) are now optional, defaulting to `0` and `false`.
- Exported PCK files now contain the Godot patch version in their header.
  - This can be used by external tools to detect the Godot version more accurately.
- Exporting a project PCK or ZIP from the command line must now be done with the new `--export-pack` command-line argument.
  - This was done to remove the ambiguity when exporting a project to macOS from the command line.
- Updated FreeType to 2.10, which changes how font metrics are calculated.
  - This may affect the appearance of some Controls, see this issue for details.
- The SCons build system now automatically detects the host platform.
  - `platform=<platform>` is no longer required when compiling for the host platform.
  - `platform=list` can be used to list the supported target platforms.
- **Windows:** Drive letters in file paths are now capitalized.
- **macOS:** Control + H and Control + D in TextEdit now delete the character at the left and right of the cursor (respectively).
- **macOS:** Command + Left in TextEdit now moves the cursor to the first non-whitespace character.
- **macOS:** Non-resizable windows are now allowed to enter fullscreen mode.
- **macOS:** The editor's title bar now uses dark mode on Mojave.

- **X11:** `OS.set_window_postion()` now takes window decorations into account.

Removed

- Unused Panel `panelf` and `panelnc` styles.
- thekla_atlas dependency, as light baking now relies on xatlas for UV unwrapping.
- Rating icons in the Asset Library, as this feature isn't implemented in the backend.
- Some editor languages are no longer available due to missing support for RTL and text shaping in Godot:
  - Affected languages are Arabic, Bengali, Persian, Hebrew, Hindi, Malayalam, Sinhalese, Tamil, Telugu and Urdu.
  - These languages will be re-added once Godot supports RTL and text shaping.
- **Android:** ARMv6 support.
- **iOS:** ARMv7 support.
  - ARMv7 export templates can still be compiled from source to support the iPhone 5 and older.

Fixed

- The Project Manager now remembers the sorting option that was previously set.
- The editor and project manager now have a minimum window size defined.
  - This prevents controls from overlapping each other by resizing the window to a very small size.
- Fixed radiance map generation, resulting in improved 3D performance and visual quality.
- Fixed issues with PBR environment mapping.
  - Materials should now look closer to what they look like in Substance Designer/Painter.
- Depth of field now affects transparent objects.
- Radiance is now generated when using a clear color sky.
- Contact shadows no longer display when shadow casting is disabled.
- Larger data types can now be constructed by swizzling in the shader language.
  - For instance, `vec2 test2 = vec2(0.0, 1.0); vec3 test3 = test2.xxx;` now works as in GLSL.
- The `AMBIENT_LIGHT_DISABLED` and `SHADOWS_DISABLED` flags now work when using the GLES2 renderer.
- The Keep background mode now works when using the GLES2 renderer.
- Several fixes to the GLES2 renderer:
  - Fixed transparency order.
  - Fixed vertex lighting being too bright.

  - Fixed occasional light flickering.
  - Fixed shadows cast from transparent materials.
  - Fog is no longer computed on unshaded materials.
    * This matches the GLES3 renderer's behavior.
  - GLES2 shader uniforms now use `highp` precision by default.
    * This prevents linking issues on some Android devices.
  - Negative OmniLights and SpotLights now work as expected.
  - The 3D editor's View Information pane now displays statistics correctly when using the GLES2 renderer.
- Textures compressed with ETC now support transparency by falling back to RGBA4444 or LA8.
- Alternate display modes are now marked as disabled in the editor when using the GLES2 renderer, as these are only supported when using GLES3.
- Fixed several inconsistencies between Particles and CPUParticles.
- Fixed particles scale randomization.
- Particles are now set to emit correctly when restarting.
- CheckBox and CheckButton now use the `check_vadjust` custom constant to adjust the icon Y position as intended.
- Fixed various issues with tab-related icons.
- Fixed issues in WebM colorspace corrections, resulting in better color output.
- CSG is now taken into account when generating navigation meshes.
- Curve2D and Curve3D interpolated values now behave as expected.
- Numeric slider grabbers in the editor inspector now update when scrolling using the mouse wheel.
- Scene modifications are no longer lost when renaming a file in the FileSystem dock.
- "Show in FileSystem" now clears the current search, so that the selected item can be seen immediately.
- LineEdit and TextEdit's context menus no longer display editing options if they are read-only.
- SpinBox mouse events are now correctly triggered by its LineEdit part.
- Per-word navigation in LineEdit and TextEdit now handles UTF-8 characters correctly.
- LineEdit placeholders, Tabs' names and WindowDialog titles now react correctly to translation changes.
- Fixed UI navigation when using gamepad analog sticks.
- Buttons' state is now reset when they exit the scene tree.
  - This prevents them from lingering in a "hovered" or "pressed" state.
- Tooltips now disappear when hiding the node they belong to.
- Encoded packet flags are no longer sent in the ENet multiplayer protocol, as ENet itself already sends that data.
  - This saves 4 bytes per packet.
- Audio trimming is now less aggressive, cutting at -50 dB instead of -30 dB.
- Audio trimming now has a small fade-out period, preventing audible pops.
- Audio mix rate and output latency settings are now consistently applied

on all platforms.
- Fixed multichannel panning for AudioStreamPlayer3D.
- Opening a recent built-in script will now load the associated scene autom-taically since doing so is required to edit the script.
- Declaring a class with `class_name` that has the same name as a singleton will now display a clearer error message.
- `script` is no longer allowed as a member variable name in GDScript, as that conflicts with the internal `script` property used by Object.
- Assigning a variable with a function index will no longer evaluate the function twice.
  - For instance, doing `a[function()] += 1` will no longer evaluate `function()` twice.
  - If the function has side effects, this may change the resulting program behavior.
- GDScript type checks are now enabled in release export templates.
- The Label font shadow now draws the font outline as well (if the base font has one).
- `Font.draw_char()` now draws the font outline as well (if the base font has one).
- The editor no longer redraws continuously when selecting a Control in a Container.
- Added some missing feature tags to the Project Settings "Override For..." menu.
- The `low_processor_mode_sleep_usec` project setting no longer affects the editor.
- Typed arrays and dictionaries no longer have their values shared across instances.
- `self` and object types can now be indexed as a dictionary again (like in Godot 3.0 and prior).
- Fixed `to_lower()` conversion with Cyrillic characters.
- The Find in Files replace dialog now allows empty replacement texts.
- The bottom panel no longer disappears when opening the theme editor on small displays.
- The script editor's color picker now changes only one color if multiple colors are present on the same line.
- The script editor's line length guideline is now drawn behind text.
- The script editor's line length guideline is now drawn at the correct position when font hinting is disabled.
- The script editor now automatically indents a line if the previous one ends with `[` or `(`.
  - This makes it possible to wrap arrays or function declarations/calls without pressing Tab every line.
- Fixed autocompletion in the script editor.
  - The script editor can now autocomplete enum values.
  - The script editor can now autocomplete node paths starting with `$"` or `$'`.

- Custom script editor templates can now use type hints.
- Shift operators with a number not between 0 and 63 (inclusive) will now result in a compile-time error in GDScript.
- Warnings no longer count towards the "Too many errors!" message.
- AnimationTrackEdit now displays invalid value keys again (as it did in 3.0).
- Fixed the display of function/audio/animation tracks in the blend tree animation filter.
- The editor shortcuts menu no longer displays all unassigned shortcuts when searching for a substring of "None".
- The editor's performance monitor now displays memory/file sizes larger than 2 GB correctly.
- The editor debugger now displays keyboard shortcuts when hovering the "Step Into", "Step Over", "Break" and "Continue" buttons.
- The editor debugger now always handles connections.
  - Subsequent connections will be dropped immediately to avoid locking.
- Large rotation offset/snap values no longer appear to be cut off in the Configure Snap dialog.
- Documentation tooltips in the editor now wrap to multiple lines correctly.
- Locked 3D nodes are no longer selectable in the 3D viewport, matching the 2D editor's behavior.
- All 3D gizmos now notify changes correctly, which means the inspector now displays up-to-date properties after using them.
- The 3D manipulator gizmo's size is now capped at low viewport heights, preventing it from outgrowing the viewport's bounds.
- The editor filesystem now refreshes on file changes if the project is located on an exFAT filesystem.
- Fixed many cases of colors not changing correctly when switching the editor from a dark theme to a light theme (or vice versa) without restarting.
- The Show in File Manager context menu option now works with files marked as favorite.
- The random number generator's seed is now properly set up.
- Antialiased and rounded StyleBoxFlat corners now handle different border widths correctly.
- The StyleBox preview now accounts for shadows and content margins.
  - This fixes the preview going out of bounds in the inspector.
- Text resources no longer contain an extraneous line break at the end of file.
- Transform's `FLIP_Y` and `FLIP_Z` constants now work as expected.
- Fixed importing BMP images.
- The positional command-line argument is now only considered to be a scene path if it ends with `.scn`, `.tscn` or `.escn`.
  - This makes it possible to parse command-line arguments in a standard fashion (`--foo bar` now works, not just `--foo=bar`).
  - This also makes it possible to use file associations or drag-and-drop and have the positional argument parsed by the project.

- The `--audio-driver` and `--video-driver` command-line arguments are now validated; an error message will be printed if an invalid value is passed.
- The `--check-only` command-line argument now returns a non-zero exit code if an invalid script is passed using `--script`.
- Exporting a project via the command-line now returns a non-zero exit code if an error occurred during exporting.
- Console output is no longer colored when standard output isn't a TTY.
  - This prevents Godot from writing ANSI escape codes when redirecting standard output or standard error to a file.
- **Android:** Gamepads are now correctly detected when the application starts.
- **Android:** Fix some keyboards being detected as gamepads and not working as a result.
- **Android:** The editor now detects if the device is connected using wireless `adb` and will debug using Wi-Fi in this case.
- **HTML5:** Fixed the pointer position on hiDPI displays.
- **HTML5:** `OS.get_system_time_msec()` now returns the correct value like on other platforms.
- **iOS:** On iOS 11 or later, gestures near screen edges are now handled by Godot instead of the OS.
- **Windows:** Line endings are now converted to CRLF when setting clipboard content.
- **Windows:** Getting the path to the Downloads directory using `OS.get_system_dir()` now works correctly.
  - This fixes line endings being invisible when pasting into other applications.
- **macOS:** `OS.get_real_window_size()` and `OS.set_window_size()` are now handled correctly on hiDPI displays.
- **X11:** `OS.get_window_position()` now returns absolute coordinates.
- **X11:** Fixed audio playing on the wrong speakers when using PulseAudio on 5.1 setups.
- **X11:** `OS.set_window_maximized()` now gives up after 0.5 seconds.
  - This makes the editor no longer freeze on startup when using fvwm.

3.1 - 2019-03-13

Added

- OpenGL ES 2.0 renderer.
- Visual shader editor.
  - New PBR output nodes.
  - Conversion between Vector3 and scalar types is now automatic.
  - Ability to create custom nodes via scripting.
  - Ports can now be previewed.
- 3D soft body physics.
- 3D ragdoll system.

- Constructive solid geometry in 3D.
- 2D meshes and skeletal deformation.
- Various improvements to KinematicBody2D.
  - Support for snapping the body to the floor.
  - Support for RayCast shapes in kinematic bodies.
  - Support for synchronizing kinematic movement to physics, avoiding an one-frame delay.
- WebSockets support using libwebsockets.
- UPnP support using MiniUPnP.
- Revamped inspector.
  - Improved visualization and editing of numeric properties.
  - Vector and matrix types can now be edited directly (no pop-ups).
  - Subresources can now be edited directly within the same inspector.
  - Layer names can now be displayed in the inspector.
  - Proper editing of arrays and dictionaries.
  - Ability to reset any property to its default value.
- Improved animation editor.
  - Simpler, less cluttered layout.
  - New Bezier, Audio and Animation tracks.
  - Several key types can be previewed directly in the track editor.
  - Tracks can now be grouped and filtered on a per-node basis.
  - Copying and pasting tracks between animations is now possible.
  - New Capture mode to blend from a node's current value to the first key in a track.
- Improved animation tree and new state machine.
  - More visual feedback in the blend tree editor.
  - 1D and 2D blend spaces are now supported.
  - Ability to write custom blending logic.
  - Support for root motion.
- New FileSystem dock.
  - Unified view of folders and files in the same panel.
  - Files can now be marked as favorites, not only folders.
  - Files now have icons representing their type, or thumbnail previews when relevant.
  - New search field to filter entries in the tree.
- OpenSimplexNoise and NoiseTexture resources.
- Optional static typing in GDScript.
  - Does not currently improve performance, but helps write more robust code.
- Warning system in GDScript.
  - Reports potential code issues such as:
    - unused variables,
    - standalone expressions,
    - discarded return values from functions,
    - unreachable code after a `return` statement,
    - ...

  - Warnings can be disabled in the Project Settings or by writing special comments.
- GDScript keyword `class_name` to register scripts as classes.
- Simple expression language independent from GDScript, used by inspector boxes that accept numeric values.
  - Can also be used in projects.
- C# projects can now be exported for Windows, Linux, and macOS targets.
- The `server` platform is back as it was in Godot 2.1.
  - It is now again possible to run a headless Godot instance on Linux.
- Support for BPTC texture compression on desktop platforms.
- New properties for SpatialMaterial.
  - Dithering-based distance fade, for fading materials without making them transparent.
  - Disable ambient light on a per-material basis.
- Option to link Mono statically on Windows.
- Unified class and reference search in the editor.
- Revamped TileSet editor with support for undo/redo operations.
- Various quality-of-life improvements to the Polygon2D and TextureRegion editors.
- RandomNumberGenerator class that allows for multiple instances at once.
- Array methods `min()` and `max()` to return the smallest and largest value respectively.
- Dictionary method `get(key[, default])` where `default` is returned if the key does not exist.
- Node method `print_tree_pretty()` to print a graphical view of the scene tree.
- String methods `trim_prefix()`, `trim_suffix()`, `lstrip()`, `rstrip()`.
- OS methods:
  - `get_system_time_msecs()`: Return the system time with milliseconds.
  - `get_audio_driver_name()` and `get_audio_driver_count()` to query audio driver information.
  - `get_video_driver_count()` and `get_video_driver_name()` to query renderer information.
  - `center_window()`: Center the window on the screen.
  - `move_window_to_foreground()`: Move the window to the foreground.
- StreamPeerTCP method `set_no_delay()` to enable the `TCP_NODELAY` option.
- EditorPlugin method `remove_control_from_container()`.
- Ability to set Godot windows as "always on top".
- Ability to create windows with per-pixel transparency.
- New GLSL built-in functions in the shader language:
  - `radians()`
  - `degrees()`
  - `asinh()`

  - `acosh()`
  - `atanh()`
  - `exp2()`
  - `log2()`
  - `roundEven()`
- New command-line options:
  - `--build-solutions`: Build C# solutions without starting the editor.
  - `--print-fps`: Display frames per second to standard output.
  - `--quit`: Quit the engine after the first main loop iteration.
- Debugger button to copy error messages.
- Support for `.escn` scenes has been added for use with the new Blender exporter.
- It is now possible to scale an OBJ mesh when importing.
- `popup_closed` signal for `ColorPickerButton`.
- Methods that are deprecated can now print warnings.
- Input actions can now provide an analog value.
- Input actions can now be mapped to either a specific device or all devices.
- DNS resolution for high-level networking.
- Servers can now kick/disconnect peers in high-level networking.
- Servers can now access IP and port information of peers in high-level networking.
- High-level multiplayer API decoupled from SceneTree (see `SceneTree.multiplayer_api/SceneTree.cust` can now be extended.
- `Input.set_default_cursor_shape()` to change the default shape in the viewport.
- Custom cursors can now be as large as 256×256 (needed to be exactly 32×32 before).
- Support for radio-looking items with icon in `PopupMenu`s.
- Drag and drop to rearrange Editor docks.
- TileSet's `TileMode` is now exposed to GDScript.
- `OS.get_ticks_usec()` is now exposed to GDScript.
- Normals can now be flipped when generated via `SurfaceTool`.
- TextureProgress bars can now be bilinear (extending in both directions).
- The character used for masking secrets in LineEdit can now be changed.
- Improved DynamicFont:
  - DynamicFonts can now use high-quality outlines generated by FreeType.
  - DynamicFonts can now have their anti-aliasing disabled.
  - DynamicFonts can now have their hinting tweaked ("Normal", "Light" or "None").
  - Colored glyphs such as emoji are now supported.
- Universal translation of touch input to mouse input.
- AudioStreamPlayer, AudioStreamPlayer2D, and AudioStreamPlayer3D now have a pitch scale property.
- Support for MIDI input.
- Support for audio capture from microphones.

- `GROW_DIRECTION_BOTH` for Controls.
- Selected tiles can be moved in the tile map editor.
- The editor can now be configured to display the project window on the previous or next monitor (relative to the editor).
  - If either end is reached, then the project will start on the last or first monitor (respectively).
- Signal in VideoPlayer to notify when the video finished playing.
- `Image.bumpmap_to_normalmap()` to convert bump maps to normal maps.
- `File.get_path()` and `File.get_path_absolute()`.
- Unselected tabs in the editor now have a subtle background for easier identification.
- The depth fog's end distance is now configurable independently of the far plane distance.
- The alpha component of the fog color can now be used to control fog density.
- The 3D editor's information panel now displays the camera's coordinates.
- New options to hide the origin and viewport in the 2D editor.
- Improved 3D editor grid:
  - The grid size and number of subdivisions can now be configured.
  - Its primary and secondary colors can now also be changed.
- Ctrl now toggles snapping in the 3D viewport.
- Find & replace in files (Ctrl + Shift + F by default).
- Batch node renaming tool (Ctrl + F2 by default).
- More editor scaling options to support HiDPI displays.
- Type icons can now be enabled in the editor again.
- Buttons in the editor to open common directories in the OS file manager:
  - project data directory,
  - user data directory,
  - user settings directory.
- Projects can now be sorted by name or modification date in the project manager.
- Projects can now be imported from ZIP archives in the project manager.
- Improved autocompletion.
  - Keywords are now present in autocompletion results.
- `editor` and `standalone` feature tags to check whether the project is running from an editor or non-editor binary.
- `android_add_asset_dir("...")` method to Android module Gradle build configuration.
- **iOS:** Support for exporting to the iPhone X.
- **iOS:** Re-added support for in-app purchases.

Changed

- Built-in vector types now use copy-on-write mode as originally intended, resulting in increased engine performance.
- The mbedtls library is now used instead of OpenSSL.

- Renamed several core files.
  - Third-party modules may have to be updated to reflect this.
- SSL certificates are now bundled in exported projects unless a custom bundle is specified.
- Improved buffer writing performance on Windows and Linux.
- Removed many debugging prints in the console.
- Export templates now display an error dialog if no project was found when starting.
- DynamicFont oversampling is now enabled by default.
- Nodes' internal logic now consistently uses internal physics processing.
- Allow attaching and clearing scripts on multiple nodes at once.
- Default values are no longer saved in scene and resource files.
- The selection rectangle of 2D nodes is now hidden when not pertinent (no more rectangle for collision shapes).
- SSE2 is now enabled in libsquish, resulting in improved S3TC encoding performance.
- Tangent and binormal coordinates are now more consistent across mesh types (primitive/imported), resulting in more predictable normal map and depth map appearance.
- Better defaults for 3D scenes.
  - The default procedural sky now has a more neutral blue tone.
  - The default SpatialMaterial now has a roughness value of 1 and metallic value of 0.
  - The fallback material now uses the same values as the default SpatialMaterial.
- Text editor themes are now sorted alphabetically in the selection dropdown.
- The 3D manipulator gizmo now has a smoother, more detailed appearance.
- The 3D viewport menu button now has a background to make it easier to read.
- QuadMeshes are now built using two triangles (6 vertices) instead of one quad (4 vertices).
  - This was done because quads are deprecated in OpenGL.
- Controls inside containers are no longer movable or resizable but can still be selected.
- The `is` GDScript keyword can now be used to compare a value against built-in types.
- Exported variables with type hints are now always initialized.
  - For example, `export(int) var a` will be initialized to `0`.
- Named enums in GDScript no longer create script constants.
  - This means `enum Name { VALUE }` must now be accessed with `Name.VALUE` instead of `VALUE`.
- Cyclic references to other scripts with `preload()` are no longer allowed.
  - `load()` should be used in at least one of the scripts instead.
- `switch`, `case` and `do` are no longer reserved identifiers in GDScript.
- Shadowing variables from parent scopes is no longer allowed in GDScript.
- Function parameters' default values can no longer depend on other param-

eters in GDScript.
- Indentation guides are now displayed in a more subtle way in the script editor.
  - Indentation guides are now displayed when indenting using spaces.
- Multi-line strings are now highlighted as strings rather than as comments in the script editor.
  - This is because GDScript does not officially support multiline comments.
- Increased the script editor's line spacing (4 pixels → 6 pixels).
- Increased the caret width in the script editor (1 pixel → 2 pixels).
- The project manager window is now resized to match the editor scale.
- The asset library now makes use of threading, making loading more responsive.
- Line spacing in the script editor, underlines and caret widths are now resized to match the editor scale.
- Replaced editor icons for checkboxes and radio buttons with simpler designs.
- Tweaked the editor's success, error, and warning text colors for better readability and consistency.
- **Android:** Custom permissions are now stored in an array and their amount is no longer limited to 20.
  - Custom permissions will have to be redefined in projects imported from older versions.
- **Android:** Provide error details when an in-app purchase fails.
- **Linux:** `OS.alert()` now uses Zenity or KDialog if available instead of xmessage.
- **Mono:** Display stack traces for inner exceptions.
- **Mono:** Bundle `mscorlib.dll` with Godot to improve portability.

Removed

- Removed the RtAudio backend on Windows in favor of WASAPI, which is the default since 3.0.
- **macOS:** Support for 32-bit and fat binaries.

Fixed

- `move_and_slide()` now behaves differently at low velocities, which makes it function as originally intended.
- AnimatedSprite2D's `animation_finished` signal is now triggered at the end of the animation, instead of as soon as the last frame displays.
- Audio buses can now be removed in the editor while they are used by AudioStreamPlayer2D/3D nodes.
- Do not show the project manager unless no project was found at all.
- The animation editor time offset indicator no longer "walks" when resizing the editor.
- Allow creation of a built-in GDScript file even if the filename suggested

already exists.
- Show tooltips in the editor when physics object picking is disabled.
- Button shortcuts can now be triggered by gamepad buttons.
- Fix a serialization bug that could cause TSCN files to grow very large.
- Gizmos are now properly hidden on scene load if the object they control is hidden.
- Camera gizmos in the 3D viewport no longer look twice as wide as they actually are.
- Copy/pasting from the editor on X11 will now work more reliably.
- `libgcc_s` and `libstdc++` are now linked statically for better Linux binary portability.
- The FPS cap set by `force_fps` in the Project Settings is no longer applied to the editor.
  - Low FPS caps no longer cause the editor to feel sluggish.
- hiDPI is now detected and used if needed in the project manager.
- The Visual Studio Code external editor option now recognizes more binary names such as `code-oss`, making detection more reliable.
- The `-ffast-math` flag is no longer used when compiling Godot, resulting in increased floating-point determinism.
- Fix spelling of `apply_torque_impulse()` and deprecate the misspelled method.
- Escape sequences like `\n` and `\t` are now recognized in CSV translation files.
- Remove spurious errors when using a PanoramaSky without textures.
- The lightmap baker will now use all available cores on Windows.
- Bullet physics now correctly calculates effective gravity on KinematicBodies.
- Setting the color `v` member now correctly sets the `s` member.
- RichTextLabels now correctly determine the baseline for all fonts.
- SpinBoxes now correctly calculate their initial size.
- OGG streams now correctly signal the end of playback.
- Android exporter no longer writes unnecessary permissions to the exported APK.
- Debugger "focus stealing" now works more reliably.
- Subresources are now always saved when saving a scene.
- Many fixes related to importers (glTF, Collada, audio), physics (Bullet), Mono/C#, GDNative, Android/iOS.
- **Mono:** Many fixes and improvements to C# support (including a `[Signal]` attribute).
- **WebAssembly:** Supply proper CORS headers.

Security

- Fixed a security issue relating to deserializing Variants.

3.0 - 2018-01-29

Added

- Physically-based renderer using OpenGL ES 3.0.
  - Uses the Disney PBR model, with clearcoat, sheen and anisotropy parameters available.
  - Uses a forward renderer, supporting multi-sample anti-aliasing (MSAA).
  - Parallax occlusion mapping.
  - Reflection probes.
  - Screen-space reflections.
  - Real-time global illumination using voxel cone tracing (GIProbe).
  - Proximity fade and distance fade (useful for creating soft particles and various effects).
  - Lightmapper for lower-end desktop and mobile platforms, as an alternative to GIProbe.
- New SpatialMaterial resource, replacing FixedMaterial.
  - Multiple passes can now be specified (with an optional "grow" property), allowing for effects such as cel shading.
- Brand new 3D post-processing system.
  - Depth of field (near and far).
  - Fog, supporting light transmittance, sun-oriented fog, depth fog and height fog.
  - Tonemapping and Auto-exposure.
  - Screen-space ambient occlusion.
  - Multi-stage glow and bloom, supporting optional bicubic upscaling for better quality.
  - Color grading and various adjustments.
- Rewritten audio engine from scratch.
  - Supports audio routing with arbitrary number of channels, including Area-based audio redirection (video).
  - More than a dozen of audio effects included.
- Rewritten 3D physics using Bullet.
- UDP-based high-level networking API using ENet.
- IPv6 support for all of the engine's networking APIs.
- Visual scripting.
- Rewritten import system.
  - Assets are now referenced with their source files, then imported in a transparent manner by the engine.
  - Imported assets are now cached in a `.import` directory, making distribution and versioning easier.
  - Support for ETC2 compression.
  - Support for uncompressed Targa (.tga) textures, allowing for faster importing.
- Rewritten export system.

  - GPU-based texture compression can now be tweaked per-target.
  - Support for exporting resource packs to build DLC / content addons.
- Improved GDScript.
  - Pattern matching using the `match` keyword.
  - `$` shorthand for `get_node()`.
  - Setters and getters for node properties.
  - Underscores in number literals are now allowed for improved readability (for example,`1_000_000`).
  - Improved performance (+20% to +40%, based on various benchmarks).
- Feature tags in the Project Settings, for custom per-platform settings.
- Full support for the glTF 2.0 3D interchange format.
- Freelook and fly navigation to the 3D editor.
- Built-in editor logging (logging standard output to a file), disabled by default.
- Improved, more intuitive file chooser in the editor.
- Smoothed out 3D editor zooming, panning and movement.
- Toggleable rendering information box in the 3D editor viewport.
  - FPS display can also be enabled in the editor viewport.
- Ability to render the 3D editor viewport at half resolution to achieve better performance.
- GDNative for binding languages like C++ to Godot as dynamic libraries.
  - Community bindings for D, Nim and Python are available.
- Editor settings and export templates are now versioned, making it easier to use several Godot versions on the same system.
- Optional soft shadows for 2D rendering.
- HDR sky support.
- Ability to toggle V-Sync while the project is running.
- Panorama sky support (sphere maps).
- Support for WebM videos (VP8/VP9 with Vorbis/Opus).
- Exporting to HTML5 using WebAssembly.
- C# support using Mono.
  - The Mono module is disabled by default, and needs to be compiled in at build-time.
  - The latest Mono version (5.4) can be used, fully supporting C# 7.0.
- Support for rasterizing SVG to images on-the-fly, using the nanosvg library.
  - Editor icons are now in SVG format, making them better-looking at non-integer scales.
  - Due to the library used, only simpler SVGs are well-supported, more complex SVGs may not render correctly.
- Support for oversampling DynamicFonts, keeping them sharp when scaled to high resolutions.
- Improved StyleBoxFlat.
  - Border widths can now be set per-corner.
  - Support for anti-aliased rounded and beveled corners.
  - Support for soft drop shadows.

- VeryLoDPI (75%) and MiDPI (150%) scaling modes for the editor.
- Improved internationalization support for projects.
  - Language changes are now effective without reloading the current scene.
- Implemented missing features in the HTML5 platform.
  - Cursor style changes.
  - Cursor capturing and hiding.
- Improved styling and presentation of HTML5 exports.
  - A spinner is now displayed during loading.
- Rewritten the 2D and 3D particle systems.
  - Particles are now GPU-based, allowing their use in much higher quantities than before.
  - Meshes can now be used as particles.
  - Particles can now be emitted from a mesh's shape.
  - Properties can now be modified over time using an editable curve.
  - Custom particle shaders can now be used.
- New editor theme, with customizable base color, highlight color and contrast.
  - A light editor theme option is now available, with icons suited to light backgrounds.
  - Alternative dark gray and Arc colors are available out of the box.
- New adaptive text editor theme, adjusting automatically based on the editor colors.
- Support for macOS trackpad gestures in the editor.
- Exporting to macOS now creates a `.dmg` disk image if exporting from an editor running on macOS.
  - Signing the macOS export now is possible if running macOS (requires a valid code signing certificate).
- Exporting to Windows now changes the exported project's icon using `rcedit` (requires WINE if exporting from Linux or macOS).
- Improved build system.
  - Support for compiling using Visual Studio 2017.
  - SCons 3.0 and Python 3 are now supported (SCons 2.5 and Python 2.7 still work).
  - Link-time optimization can now be enabled by passing `use_lto=yes` to the SCons command line.
    - Produces faster and sometimes smaller binaries.
    - Currently only supported with GCC and MSVC.
  - Added a progress percentage when compiling Godot.
  - `.zip` archives are automatically created when compiling HTML5 export templates.
- Easier and more powerful way to create editor plugins with EditorPlugin and related APIs.

Changed

- Increased the default low-processor-usage mode FPS limit (60 → 125).
  - This makes the editor smoother and more responsive.
- Increased the default 3D editor camera's field of view (55 → 70).
- Increased the default 3D Camera node's field of view (65 → 70).
- Changed the default editor font (Droid Sans → Noto Sans).
- Changed the default script editor font (Source Code Pro → Hack)
- Renamed `engine.cfg` to `project.godot`.
  - This allows users to open a project by double-clicking the file if Godot is associated to `.godot` files.
- Some methods from the `OS` singleton were moved to the new `Engine` singleton.
- Switched from GLEW to GLAD for OpenGL wrapping.
- Changed the SCons build flag for simple logs (`colored=yes` → `verbose=no`).
- The HTML5 platform now uses WebGL 2.0 (instead of 1.0).
- Redesigned the Godot logo to be more legible at small sizes.

Deprecated

- `opacity` and `self_opacity` are replaced by `modulate` and `self_modulate` in all 2D nodes, allowing for full color changes in addition to opacity changes.

Removed

- Skybox support.
  - Replaced with panorama skies, which are easier to import.
- Opus audio codec support.
  - This is due to the way the new audio engine is designed.
- HTML5 export using asm.js.
  - Only WebAssembly is supported now, since all browsers supporting WebGL 2.0 also support WebAssembly.