138 → 139

- `CubeUVRefractionMapping` and `MeshStandardMaterial.refractionRatio` have been removed. Use the transmission related properties of `MeshPhysicalMaterial` if you want to model refraction with a PBR material.

137 → 138

- `WebGLMultisampleRenderTarget` has been removed. To use multisampling as before, use `WebGLRenderTarget` and set the new `samples` property to a value greater `0`.
- The node material in `examples/jsm/nodes` has been replaced with a new implementation.
- `ColladaLoader`, `KMZLoader` and `PLYLoader` require a sRGB workflow now.
- `OBJExporter`, `ColladaExporter` and `PLYExporter` produce assets with sRGB encoded colors now.
- `VRMLoader` has been removed. Use three-vrm instead.
- The second argument of `GLTFLoader`'s `parser.loadTextureImage()` has been changed from image source definition to image source index.
- `Euler.toVector3()` has been removed. Use `Vector3.setFromEuler()` instead.
- `DataTexture3D` has been renamed to `Data3DTexture`.
- `DataTexture2DArray` has been renamed to `DataArrayTexture`.
- `WebGLRenderTarget.setTexture()` has been removed.

136 → 137

- `WebGLRenderer` now creates the WebGL context with an alpha channel regardless of the value of `alpha` passed in the constructor. However, the value of `alpha` is still used by the renderer when clearing the context first thing every frame.
- `RGBFormat` has been removed. Please use `RGBAFormat` instead.
- `RGBIntegerFormat` has been removed. Please use `RGBAIntegerFormat` instead.
- `UnsignedShort565Type` has been removed. Please use `UnsignedShort5551Type` instead.
- `BasisTextureLoader` has been deprecated. Please use `KTX2Loader` instead.
- The `SRGB8_ALPHA8_ASTC*` texture formats have been removed. If you want to use sRGB ASTC formats, use the regular `RGBA_ASTC_*` formats and set the `encoding` texture property to `sRGBEncoding`.
- With WebGL 2 uncompressed sRGB encoded textures have to use the `RGBAFormat` with `UnsignedByteType` now.
- `RoughnessMipmapper` has been removed.
- `Material.format` has been removed.
- `GLTFExporter` does not support `RGBFormat` anymore. Please use `RGBAFormat` instead.
- The default texture format of `VideoTexture` is now `RGBAFormat` (instead of `RGBFormat`).
- sRGB decode in GLSL has been removed. sRGB texture always have to use `RGBAFormat` + `UnsignedByteType` format now.
- The ES6 import of `three.js` and `examples/jsm` modules in web sites now requires the usage of an import map to resolve the `three` bare import specifier.
- The ES6 import of `examples/jsm` and `examples/fonts` in certain bundlers like esbuild now requires an extension on filenames.
- `OBJ/MTLLoader` requires a sRGB workflow now.
- Changing `Material.transparent` after its initial use requires setting `needsUpdate` to `true`.

135 → 136

- HDR workflows with WebGL 1 now require half float texture extension support (`OES_texture_half_float` / `OES_texture_half_float_linear`).
- If you create an instance of `DataTexture` , `DataTexture2DArray` or `DataTexture3D` , you have to set `needsUpdate` to `true` as soon as the texture data are ready.
- `WebGLRenderer.copyFramebufferToTexture()` has to be used with the new class `FramebufferTexture` now.
- ETC1 texture compression can now only be used with a WebGL 1 rendering context.
- The `keydown` event listener of `ArcballControls` has been removed. You have to implement it on app level if necessary.
- `ArcballControls.setTarget()` has been removed. Update the `target` property instead.
- When changing `WebGLRenderer.toneMapping` , it is no longer necessary to set `Material.needsUpdate` to `true` .
- `EXRLoader` no longer supports the data type `UnsignedByteType` . Use the default data type `HalfFloatType` instead.
- `PMREMGenerator` now uses half float render targets internally, and no longer uses `RGBEEncoding` .
- `RGBM7Encoding` and `RGBM16Encoding` have been removed. It is still possible to load RGBM texture as FP16 or FP32 textures. The default `type` is `HalfFloatType` .
- `RGBEEncoding` and `RGBEFormat` have been removed. `RGBELoader` and `HDRCubeTextureLoader` no longer support the data type `UnsignedByteType` . It is still possible to load RGBE texture as FP16 or FP32 textures. The default `type` is `HalfFloatType` .
- `RGBDEncoding` has been removed.
- `WebGLRenderer.gammaFactor` and `THREE.GammaEncoding` have been removed. Please use post processing (a gamma correction pass) if you need a special gamma color space.

134 → 135

- `dat.gui` has been replaced with `lil-gui` .
- The dimensions, format and type of a texture cannot be changed after the initial use now.
- `GLTFExporter.parse()` has a new signature. The third parameter is now an error callback. The exporter options are now passed in as the fourth argument.
- `LogLuvEncoding` has been removed. Please use the new `LogLuvLoader` for loading Logluv TIFF HDR images as (half precision) floating point textures.

133 → 134

- `DeviceOrientationControls` has been removed.
- `ImmediateRenderObject` has been removed.
- `OrbitControls` no longer supports zooming (via mouse wheel) while rotating.
- `FileLoader` now uses `fetch` instead of `XMLHttpRequest` .

132 → 133

- The `recursive` parameter of `Raycaster.intersectObject()` and `Raycaster.intersectObjects()` is now `true` by default.
- Some default values of `ExtrudeGeometry` 's parameters have changed. `depth` is now `1` , `bevelThickness` is now `0.2` and `bevelSize` is now `bevelThickness - 0.1` .

- `ParametricGeometry` has been removed from core. It is now located in `examples/jsm/geometries/ParametricGeometry.js`.
- `TextGeometry` has been removed from core. It is now located in `examples/jsm/geometries/TextGeometry.js`.
- `FontLoader` and `Font` have been removed from core. Both classes are now located in `examples/jsm/loaders/FontLoader.js`.

131 → 132

- `BufferGeometryUtils` is now imported using this pattern: `import * as BufferGeometryUtils from './jsm/utils/BufferGeometryUtils.js';`
- `KTX2Loader` requires an updated version of the Basis Universal transcoder from `examples/js/libs/basis`
- `MeshPhysicalMaterial.sheen` has been renamed to `sheenTint`.
- `BufferGeometry.computeFaceNormals()` has been removed. The method was not implemented and just a stub. Calling it did not affect the geometry.
- `MeshStandardMaterial` and `MeshPhysicalMaterial` shaders are now more correct and may result in less shiny renders for models using rough materials.

130 → 131

- The `morphTargets` and `morphNormals` property of materials has been removed.
- `MeshStandardMaterial.vertexTangents` has been removed.
- The default `widthSegments` and `heightSegment` properties of `SphereGeometry` have been increased to `32` and `16`.
- The default type of textures loaded with `RGBELoader`, `HDRCubeTextureLoader` and `EXRLoader` is now `THREE.HalfFloatType`.
- The `envMap` property of `MeshStandardMaterial` and `MeshPhysicalMaterial` is now internally converted to a PMREM. It might be necessary to update the scene's lighting if no PMREM was previously used.

129 → 130

- Controls no longer call `preventDefault()` on pointer and mouse events.
- `DragControls`, `OrbitControls` and `TrackballControls` now use pointer events for touch interaction.

128 → 129

- The backward compatibility for the deprecated third and forth arguments `renderTarget` and `forceClear` of `WebGLRenderer.render()` has been removed.
  - Use `WebGLRenderer.setRenderTarget()` and `WebGLRenderer.clear()` instead.
- The `skinning` property of materials has been removed.
  - The same material can now be reused between `Mesh` and `SkinnedMesh`.
- `Matrix4.makeShear()` has a new signature. Please make a note of it.

127 → 128

- All central engine components like `Object3D`, `BufferGeometry` or `ShaderMaterial` are now ES6 classes.

    - This might impact your application if you derive custom classes from `three.js` classes via ES5 syntax. In this case, you have to upgrade your code to ES6 or transpile `three.js` to ES5.
- The JavaScript code in `examples/js` generated from modules in `examples/jsm` is now based on ES6.
    - If you need ES5 code in your project, update the configuration in .babelrc.json and regenerate `examples/js` by using `npm run build-examples`. However, the better solution is to upgrade your code to ES6.
- NPM: ES6 modules in `examples/jsm` now import using the bare specifier `three`.
    - This change breaks working with modules in cdns such as https://www.jsdelivr.com/ and https://unpkg.com/. Please use https://www.skypack.dev/ instead.
- `XLoader` has been removed.

r126 → r127

- Controls no longer use `stopPropagation()` in their event listeners. In certain use cases event listeners on application level are now able to process events (which was not possible before).
- `Scene.background` no longer supports instances of `WebGLCubeRenderTarget`. Please assign the `texture` property of render targets.
- `WebGLRenderer.setFramebuffer()` has been removed.
- `AssimpLoader` has been removed.
- `Plane.intersectLine()` now returns `null` when no intersection was found.
- `WebGLRenderer.maxMorphTargets` and `WebGLRenderer.maxMorphNormals` have been removed.
- When using `EventDispatcher`, `event.target` is only valid for the duration of the listener callback now.

r125 → r126

- TypeScript type declaration files have been moved to three-types/three-ts-types.
- `Face3` has been removed from core. It is now located in `examples/jsm/deprecated/Geometry.js`.
- `Ocean` and `OceanShaders` have been removed.
- `polyfills.js` has been removed. The polyfills for `Array.from()`, `Function.name`, `Number.EPSILON`, `Number.isInteger`, `Math.log2`, `Math.sign` and `Object.assign()` need to be added at application level for IE11 support: misc_legacy.
- `CSS3DRenderer` no longer supports IE11.
- `WebGLRenderer` now sets `gl.UNPACK_COLORSPACE_CONVERSION_WEBGL` to `gl.NONE`. Embedded color space data (ICC-profiles) in texture images will now be ignored.

r124 → r125

- `Geometry` has been removed from the core. It is now located in `examples/jsm/deprecated/Geometry.js`.
- Geometry generators like `BoxGeometry` now produce a `BufferGeometry`.
- `Mesh`, `Line` and `Points` no longer support raycasting with `Geometry`.
- `Line.computeLineDistances()` no longer supports `Geometry`.
- Exporters no longer support `Geometry`.
- `DecalGeometry`, `EdgesGeometry`, `WireframeGeometry`, `Projector`, `LineGeometry`, `LineSegmentsGeometry`, `ConvexHull`, `EdgeSplitModifier` and `TessellateModifier` no longer support `Geometry`.

- `ConvexBufferGeometry` has been removed. Similar to `DecalGeometry`, `ConvexGeometry` is now derived from `BufferGeometry`.
- `TeapotBufferGeometry` has been renamed to `TeapotGeometry`.
- `RoundedBoxBufferGeometry` has been renamed to `RoundedBoxGeometry`.
- `3MFLoader`, `AMFLoader`, `EXRLoader`, `FBXLoader`, `KMZLoader`, `NRRDLoader`, `TiltLoader` and `VTKLoader` now depend on `fflate`. Other decompression libs (`JSZip` and `ZLib`) are no longer used.
- `SubdivisionModifier` has been removed.
- `SimplifyModifier` no longer supports `Geometry` and now relies on `BufferGeometryUtils`.
- `OBJLoader2` has been removed. If you still need it, use this repository.
- `OrbitControls` no longer listens to key events by default. You have to call `OrbitControls.listenToKeyEvents()` if your app requires key controls.
- `BufferGeometryUtils.computeTangents()` has been moved into the core. You can now use `BufferGeometry.computeTangents()`.
- `RectAreaLightHelper.update()` has been removed.

r123 → r124

- `ColladaLoader` stores animation clips in `collada.scene.animations` now.
- `WebGLRenderer.getClearColor()` now expects a target argument.
- `TypedArrayUtils` and `webgl_nearestneighbour` demo have been removed.
- The converter scripts `fbx2three` and `obj2three` have been removed.

r122 → r123

- `Matrix3/4.getInverse()` has been deprecated. Please use the new method `.invert()` with this pattern: `matrixInv.copy( matrix ).invert();`.
- `Quaternion.inverse()` has been renamed to `Quaternion.invert()`.
- The option `forcePowerOfTwoTextures` has been removed from `GLTFExporter`.
- The first parameter of `DRACOExporter.parse()` is now of type `Mesh` or `Points`.
- `DragControls` now use Pointer Events. This change might require that you move your custom event listeners to Pointer Events, too.

r121 → r122

- `ExplodeModifier` has been removed.
- `Fire` and the respective `webgl_fire` example have been removed.

r120 → r121

- The `detail` parameter of `PolyhedronGeometry` is now more fine-grained. Meaning it can produce now more different subdivisions.
- `LightShadow` has been removed from the public API. If you need to configure a custom shadow frustum for spot lights, use the new `SpotLightShadow.focus` property.
- Geometry generator classes are now located in their own files. For example `BoxGeometry` and `BoxBufferGeometry` no longer share a single file and are now located in `src/geometries/BoxGeometry.js` and `src/geometries/BoxBufferGeometry.js`.
- `WebGLCubeRenderTarget.texture` is now of type `CubeTexture`.

- `TimelinerController` and the respective example `misc_animation_authoring` have been removed.
- `TypedGeometryExporter` has been removed.
- A performance improvement for `DRACOLoader` required an update of the DRACO library.

r119 → r120

- `Scene.dispose()` has been removed.
- `WebGLRenderTarget.stencilBuffer` and `WebGLCubeRenderTarget.stencilBuffer` are now false by default. Enable the buffer by setting `stencilBuffer: true` in the constructor options.
- When using `ShaderMaterial` and `RawShaderMaterial`, it's now necessary to set the new `glslVersion` property to `THREE.GLSL3` if you want to write GLSL 3.0 shader code. The GLSL version directive is not allowed in custom shader code anymore. It's always added automatically by the engine.
- `SpotLightShadow` and `DirectionalLightShadow` have been removed from the public API.
- `CannonPhysics` has been removed.
- `OrbitControls`, `TrackballControls` and `TransformControls` now use Pointer Events. This change might require that you move your custom event listeners to Pointer Events, too.

r118 → r119

- `MeshPhysicalMaterial.transparency` has been renamed to `MeshPhysicalMaterial.transmission`.

r117 → r118

- `SphericalReflectionMapping` is no longer supported. Consider using a Matcap texture with `MeshMatcapMaterial` instead.
- `WebGLRenderer.toneMappingWhitePoint` has been removed.
- `Uncharted2ToneMapping` has been removed. However, it's now possible to define your own custom tone mapping function by using `CustomToneMapping`. Check out the tone mapping example for more information.
- `WebGLRenderer` automatically creates a WebGL 2 rendering context now (and fallbacks to WebGL 1 if necessary). If your project can only use WebGL 1, you can use WebGL1Renderer.
- The default value of `OrbitControls.screenSpacePanning` is now `true`.
- `Water` can only be used when setting `WebGLRenderer.outputEncoding` to `THREE.LinearEncoding`.
- `shininess`, `specular` and `specularMap` have been removed from `MeshToonMaterial`.

r116 → r117

- The TypeScript declaration for `Texture.mipmaps` is now `any[]` instead of `ImageData[]`.
- `InstancedBufferGeometry.maxInstancedCount` has been renamed to `InstancedBufferGeometry.instanceCount`.
- The constructor of `CubeCamera` has changed. It now expects an instance of WebGLCubeRenderTarget as third parameter. The fourth parameter `options` has been removed.

r115 → r116

- The default value of `WebGLRenderer.toneMapping` is now `NoToneMapping`.
- `Sphere.empty()` has been renamed to `Sphere.isEmpty()`.

- `TranslucentShader` has been renamed to `SubsurfaceScatteringShader`.
- `PDBLoader` no longer returns raw bonds data in the JSON result.
- The `options` parameter of `VRButton.createButton()` has been removed. Please set the reference space type via `WebGLRenderer.xr.setReferenceSpaceType()` instead.

r114 → r115

- The `throwOnDegenerate` parameter of `Matrix3.getInverse()` and `Matrix4.getInverse()` has been removed. In addition, the methods now return the zero matrix if one tries to invert a matrix having a determinant of zero.
- The TypeScript declaration for `Geometry.boundingBox` and `Geometry.boundingSphere` is now nullable, as it actually be.
- The shader syntax `#pragma unroll_loop` is now deprecated. Use `#pragma unroll_loop_start` / `end` instead.

r113 → r114

- `Material.vertexColors` is now a boolean. The default value is `false`.
- `Raycaster` honors now invisible 3D objects in intersection tests. Use the new property Raycaster.layers for selectively ignoring 3D objects during raycasting.
- `GLTFLoader` now returns an instance of `Group` instead of `Scene`.
- `GLTFLoader` now sets `depthWrite` to `false` for transparent materials.
- The `OBJ` and `FBX` converters now require the esm npm package.

r112 → r113

- `Math` has been renamed to `MathUtils`, and `/examples/js/utils/MathUtils.js` has been promoted to the core.
- `WebGLRenderTargetCube` has been renamed to `WebGLCubeRenderTarget`, and the constructor signature is now `WebGLCubeRenderTarget( size, options )`.
- `Geometry.applyMatrix()` has been renamed to `Geometry.applyMatrix4()`.
- `BufferGeometry.applyMatrix()` has been renamed to `BufferGeometry.applyMatrix4()`.
- `Object3D.applyMatrix()` has been renamed to `Object3D.applyMatrix4()`.
- `LineSegmentsGeometry.applyMatrix()` has been renamed to `LineSegmentsGeometry.applyMatrix4()`.
- `Frustum.setFromMatrix()` has been renamed to `Frustum.setFromProjectionMatrix()`.
- `RaytracingRenderer` has been removed.
- `WebGLDeferredRenderer` has been removed.
- `GammaCorrectionShader` converts to `sRGB` now.
- The color of the default material for `Mesh`, `Points`, `Line`, and all derived classes, is now white.

r111 → r112

- `PMREMGenerator` has a new implementation and is now part of the core library. Check out the webgl_loader_gltf example to understand the new workflow.
- `WebGLRenderer.gammaInput` has been removed. Set the encoding for textures via `Texture.encoding` instead.
- `WebGLRenderer.gammaOutput` has been removed. Please use `WebGLRenderer.outputEncoding` instead.
- `MeshToonMaterial` does not support environment maps anymore.

- `Mesh.drawMode` and `Mesh.setDrawMode()` have been removed. `WebGLRenderer` does render meshes always with `THREE.TrianglesDrawMode` now. Please use `BufferGeometryUtils.toTrianglesDrawMode()` to transform `THREE.TriangleStripDrawMode` and `THREE.TriangleFanDrawMode` of existing geometries to `THREE.TrianglesDrawMode`.
- `TerrainShader`, `SkinShader` and `CarControls` have been removed.
- `WebVR` support has been removed. Please use `WebXR` instead.
- The default value of `MeshStandardMaterial.roughness` has changed from `0.5` to `1`.
- The default value of `MeshStandardMaterial.metalness` has changed from `0.5` to `0`.
- `FaceNormalsHelper`, `LightProbeHelper`, `PositionalAudioHelper`, `RectAreaLightHelper`, `VertexNormalsHelper` and `VertexTangentsHelper` are now part of the examples.
- Instances of `BufferGeometry` require at least a position attribute *or* index now.

r110 → r111

- The semantics of `Material.needsUpdate` has changed. Setting it to `true` now increases the internal version counter (similar to `Texture` or `BufferAttribute`). It's not possible anymore to use `Material.needsUpdate` in conditional statements.
- `LegacyGLTFLoader` and `LegacyJSONLoader` have been removed.
- `WebVRManager.setPoseTarget()` has been removed.
- `WebVRManager` and `WebXRManager` do no longer modify the camera when not presenting.
- The default value of `Ray.direction` is now `( 0, 0, - 1)`.
- Instances of `BufferGeometry` require at least a position attribute now.

r109 → r110

- `BufferAttribute.dynamic` and `BufferAttribute.setDynamic()` have been deprecated. Please use `BufferAttribute.usage` and `BufferAttribute.setUsage()` instead.
- `BufferGeometry.addAttribute()` has been renamed to `BufferGeometry.setAttribute()`.
- `BufferGeometry.removeAttribute()` has been renamed to `BufferGeometry.deleteAttribute()`.
- `CubemapGenerator` has been removed. Please use `WebGLRenderTargetCube.fromEquirectangularTexture()` instead.
- `EquirectangularToCubeGenerator` has been removed. Please use `WebGLRenderTargetCube.fromEquirectangularTexture()` instead.
- The second constructor parameter `domElement` of `OrbitControls`, `TrackballControls`, `TransformControls`, `FlyControls`, `PointerLockControls` and `FirstPersonControls` is now mandatory.
- `OrbitControls` and `TrackballControls` do not support `document` as an argument for `domElement` anymore. Please use the canvas element of the renderer (`renderer.domElement`) instead.
- `Audio.startTime` has been removed. Please use `Audio.play( delay )` instead.
- When loading a `DataTexture` via `DataTextureLoader`, it's default `minFilter` value is now `LinearFilter`.
- `AssimpJSONLoader` has been removed. Please use `AssimpLoader` instead.
- `SoftwareRenderer` has been removed.

r108 → r109

- `Loader.Handler` has been removed. Use `LoadingManager`'s `.addHandler()`, `.removeHandler()` and `.getHandler()` instead.
- `BabylonLoader` has been removed. Please use `glTF` instead.
- `PlayCanvasLoader` has been removed. Please use `glTF` instead.
- `AWDLoader` has been removed. Please use `glTF` instead.
- `SEA3DLoader` has been removed. Please use `glTF` instead.
- `EditorControls` is now located in `editor/js`.
- `OrthographicTrackballControls` has been removed. `TrackballControls` now supports orthographic cameras.
- `BufferAttribute.setArray()` has been removed.
- Displacement maps do not ignore the transformation of texture coordinates anymore.
- It's not necessary anymore to set `.needsUpdate` to `true` when creating a `DataTexture` (assuming the data are provided at construction time as a constructor parameter).
- `BoxGeometry` and `BoxBufferGeometry` are now ES6 classes (except in `three.js` and `three.min.js`).

r107 → r108

- `CTMLoader` has been removed.
- In `MeshPhysicalMaterial`, renamed `.clearCoat` to `.clearcoat` and `.clearCoatRoughness` to `.clearcoatRoughness`.
- Removed `.initMaterials()` and `.createMaterial()` from `Loader`.
- The obsolete callbacks `onLoadStart()`, `onLoadProgress()` and `onLoadComplete()` have been removed from `Loader`.
- `DRACOLoader.setDecoderPath()` and `DRACOLoader.setDecoderConfig()` are now instance methods.

r106 → r107

- In the Texture Filter Constants, `MipMap` is now `Mipmap`. For example, `THREE.LinearMipMapLinearFilter` is now `THREE.LinearMipmapLinearFilter`.
- Renamed `WebGLRenderer.getActiveMipMapLevel()` to `WebGLRenderer.getActiveMipmapLevel()`.
- `WEBGL` (the namespace from `examples/js/WebGL.js`) is now in the `THREE` namespace.
- `WEBVR` (the namespace from `examples/js/vr/WebVR.js`) is now in the `THREE` namespace.
- The module `MapControls` is now part of `OrbitControls`. Check out the official example for more information.
- `OrbitControls` and `MapControls` now have a new default value for `dampingFactor`.
- `WebGLRenderer.context` has been removed. Please use `WebGLRenderer.getContext()` instead.
- `FBXLoader` now correctly sets the texture encoding. When using FBX assets in your scene, you have to set `renderer.gammaOutput = true;` (unless you need post-processing in linear colorspace).
- When loading an FBX asset with TGA textures, `FBXLoader` requires now the following setup: `THREE.Loader.Handlers.add( /\.tga$/i, new TGALoader() );`.

r105 → r106

- All examples now use ES6 modules.
- `VRMLLoader` has a new implementation. It's necessary now to include `chevrotain.min.js` into your code. Check out the official example for more details.

- The optional `update` arg has been removed from the public API of the following methods: `Euler.setFromRotationMatrix( m, order )`, `Euler.setFromQuaternion( q, order )`, and `Quaternion.setFromEuler( e )`.
- `GPUParticleSystem` has been removed.
- `DracoExporter` has been renamed to `DRACOExporter`.
- Objects of type `LOD` are now updated automatically by `WebGLRenderer`. Set `LOD.autoUpdate` to `false` if you want to perform the update by yourself.
- MTL related functions like `.loadMtl()` have been removed from `OBJLoader2`. Please use `MTLLoader` and `MtlObjBridge` as shown in basic obj2 example.
- `OBJLoader2` has been removed from `examples/js/loaders`. Please use the module version in `examples/jsm/loaders`.

r104 → r105

- `WebGLRenderer.debug.checkShaderErrors` is now `true` by default.
- `EffectComposer.setSize()` now respects the pixel ratio. An instance of `EffectComposer` can now be resized with the same `width` and `height` values like `WebGLRenderer`.
- Renamed `QuickHull` to `ConvexHull`. The file is now located in `examples/js/math`.
- `SimplexNoise` and `ImprovedNoise` are now in the `THREE` namespace and located in `examples/js/math`.
- `AnimationClipCreator` and `TimelinerController` are now located in `examples/js/animation`.
- `ParametricGeometries` is now located in `examples/js/geometries`.
- `hilbert2d` and `hilbert3D` were removed. Please use `GeometryUtils.hilbert2D()` and `GeometryUtils.hilbert3D()` instead.

r103 → r104

- For performance reasons, `WebGLRenderer` does no longer perform error checking and reporting when shader programs are being compiled. You have to set `renderer.debug.checkShaderErrors` to `true` to restore the previous behavior.
- `Object3D.applyMatrix()` now updates the local matrix if `Object3D.matrixAutoUpdate` is set to `true`.

r102 → r103

- The `npm` script `npm run editor` was removed. The editor is now a Progressive Web App (PWA).
- The callback parameter of `SVGLoader.onLoad()` is now an object (`data`) containing the root node of the SVG document and an array of `ShapePath` objects. Also, all paths are returned now (not only the ones with `fill` color)
- Removed `.allocTextureUnit()`, `.setTexture2D()`, `.setTexture()` and `.setTextureCube()` from `WebGLRenderer`. These methods were never intended to be part of `WebGLRenderer`'s public API and are now private (as a part of `WebGLTexture`).

r101 → r102

- Removed `renderTarget` and `forceClear` parameters from `WebGLRenderer.render()`. Please use `.setRenderTarget()` and `.clear()` instead before you perform the rendering. Be aware that

it's now necessary to execute `renderer.setRenderTarget( null )` in order to unset an active render target.
- Removed `.activeCubeFace` and `.activeMipMapLevel` from `WebGLRenderTargetCube`. They are now parameters of `WebGLRenderer.setRenderTarget()`.
- In `WebGLRenderer.setViewport()` and `WebGLRenderer.setScissor()`, `(x, y)` is the coordinate of the *lower left* corner of the rectangular region.
- `WebGLRenderer.getSize()` now requires a `Vector2` argument.
- `WebGLRenderer.getCurrentViewport()` now requires a `Vector4` argument.

r100 → r101

- Added `FirstPersonControls.lookAt()`. `lat`, `lon`, `phi`, `theta` and `target` were removed from the public API. `FirstPersonControls` also respects the initial camera orientation now.
- `MeshStandardMaterial` and `MeshPhysicalMaterial` now preserve energy for IBL lighting, resulting in brighter, more accurate colors for metallic materials with high roughness values when lit via a map generated by PMREMGenerator.

r99 → r100

- `Octree` has been removed.
- Removed `Geometry` support from `Mesh.updateMorphTargets()`. Use `BufferGeometry` instead.
- The default orientation of `RectAreaLight` has changed. It now looks along the negative z-axis.

r98 → r99

- `WebGLRenderTarget.texture.generateMipmaps` is now set to `false` by default.
- There is a new (not backwards compatible) implementation for `SSAOShader` and `SSAOPass`.
- `JSONLoader` has been removed from core. It is now located in `examples/js/loaders/deprecated/LegacyJSONLoader.js`.
- Removed `Geometry` support from `ObjectLoader`. You have to include `LegacyJSONLoader` if you still want to load geometry data of type `Geometry`.
- Removed `Geometry` support from `SkinnedMesh`. Use `BufferGeometry` instead.
- Removed `SkinnedMesh.initBones()`. The `SkinnedMesh` constructor does not build the bone hierarchy anymore. You have to do this by yourself and then call SkinnedMesh.bind() in order to bind the prepared skeleton.

r97 → r98

- Renamed `ObjectLoader.setTexturePath()` to `ObjectLoader.setResourcePath()`. Added `ObjectLoader.setPath()`.
- `CanvasRenderer` has been removed.
- The order of `LoadingManager`'s callbacks has changed. `onError()` is now called before `onLoad()`.

r96 → r97

- Removed `BinaryLoader`.
- `WebGLRenderer.clearTarget()` is now deprecated. Use `WebGLRenderer.setRenderTarget()` in combination with `WebGLRenderer.clear()` instead.
- Renamed `JSONLoader.setTexturePath()` to `JSONLoader.setResourcePath()`.
- Renamed `MTLLoader.setTexturePath()` to `MTLLoader.setResourcePath()`.

- `GLTFLoader.setPath()` is now used for the original glTF file. Use `GLTFLoader.setResourcePath()` if you want to change the path for resources like textures.
- `TDSLoader.setPath()` is now used for the original 3DS file. Use `TDSLoader.setResourcePath()` if you want to change the path for resources like textures.
- Refactored `PointerLockControls`. Please have a look at the official example to see the new API.
- `Detector.js` was refactored to `WebGL.js`.

r95 → r96

- `Object3D.lookAt()` now supports rotated parents.
- `EquirectangularToCubeGenerator` constructor args have changed.

r94 → r95

- `OrbitControls.mouseButtons` key-value pairs have been renamed. Please make a note of it if you wish to change the mouse button bindings.
- `BufferSubdivisionModifier` has been removed. Use `SubdivisionModifier` instead.
- Sprites are now rendered concurrently with opaque and transparent objects.
- Keyframe tracks are no longer automatically validated and optimized. Users need to explicitly call `.validate/optimize()`.
- Renamed shader chunk `lights_pars_maps` to `envmap_physical_pars_fragment`.

r93 → r94

- `TDSLoader` now produces `BufferGeometry`.
- `MD2Loader` now produces `BufferGeometry`.
- `XLoader` now produces `BufferGeometry`.
- Removed deprecated CTM, FBX, msgpack and UTF8 converters.
- Removed deprecated `UTF8Loader`.
- Renamed `EquiangularToCubeGenerator` to `EquirectangularToCubeGenerator`.
- Removed deprecated `VRControls` and `VREffect`.
- `DaydreamController` and `GearVRController` are now deprecated. The new directory of these file is `examples/js/vr/deprecated/`.

r92 → r93

- Renamed option `amount` to `depth` in `ExtrudeBufferGeometry`.
- The Blender exporter has been removed. See #12903 and #14117 for more information. Also have a look at the new guide Loading 3D models.
- `STLBinaryExporter` has been removed. It's now part of `STLExporter`.
- Renamed `WebGLRenderer.animate()` to `WebGLRenderer.setAnimationLoop()`.

r91 → r92

- Removed option `frames` from `ExtrudeBufferGeometry`.
- Removed `.getArrays()` from `ExtrudeBufferGeometry`.
- Removed `.addShapeList()` from `ExtrudeBufferGeometry`.
- Removed `.addShape()` from `ExtrudeBufferGeometry`.
- `ExtrudeGeometry.WorldUVGenerator` is now private.
- `SVGLoader` now parses SVG input and returns an array of `ShapePath` objects.

r90 → r91

- `Geometry.center()` and `BufferGeometry.center()` now return `this` instead of `offset`.
- `optionalTarget`s are now mandatory method parameters (with exception of curve classes).
- Split `ShaderChunk.lights_pars` into `ShaderChunks.lights_pars_begin` and `ShaderChunks.lights_pars_maps`.
- Split `ShaderChunk.lights_template` into `ShaderChunks.lights_fragment_begin`, `ShaderChunnks.lights_fragment_maps` and `ShaderChunks.lights_fragment_end`.
- Split `ShaderChunk.normal_fragment` into `ShaderChunks.normal_fragment_begin` and `ShaderChunks.normal_fragment_maps`.
- The semantics of `AnimationAction.repetition` has changed. The first run of the animation is now taken into account.
- Removed `copyIndicesArray()` from `BufferAttribute`.
- Removed `getWorldRotation()` from `Object3D`.
- Renamed `Triangle.area()` to `Triangle.getArea()`.
- Renamed `Triangle.barycoordFromPoint()` to `Triangle.getBarycoord()`.
- Renamed `Triangle.midpoint()` to `Triangle.getMidpoint()`.
- Renamed `Triangle.normal()` to `Triangle.getNormal()`.
- Renamed `Triangle.plane()` to `Triangle.getPlane()`.
- Removed options `material` and `extrudeMaterial` from `ExtrudeGeometry`.
- Removed `vertices` from `renderer.info.render`.
- BasicDepthPacking: Depth values at the near plane are now encoded as white. Depth values at the far plane as black.

r89 → r90

- `Lensflare` has been moved out of the core. Please use examples/js/objects/Lensflare.js if you need lens flares in your scene. Also have a look at the official example to see the new usage of `Lensflare`.
- `SceneUtils` has been moved out of the core. It is now located at examples/js/utils/SceneUtils.js.
- Removed `.shadowMap.renderReverseSided` from `WebGLRenderer`. Set `Material.shadowSide` instead.
- Removed `.shadowMap.renderSingleSided` from `WebGLRenderer`. Set `Material.shadowSide` instead.
- Removed `.setFaceCulling()` from `WebGLRenderer`.
- Removed the JSON exporters for Maya and 3ds Max.
- Removed `.computeLineDistances()` from `Geometry`. Use `Line.computeLineDistances()` instead.

r88 → r89

- `ImageUtils` has been removed.
- Removed `extractUrlBase()` from `Loader`. Use `LoaderUtils.extractUrlBase()` instead.
- `ShapeUtils.triangulateShape()` uses a new and more robust polygon triangulation algorithm now.
- `ShapeUtils.triangulate()` has been removed.
- `Reflector`, `Refractor`, `Water` and `Water2` accept now any planar geometry and not only rectangular ones. The respective constructors have a new signature.

r87 → r88

- `CombinedCamera` has been removed.
- `ColladaLoader2` has replaced `ColladaLoader`.
- `VRMLLoader` now produces `BufferGeometry`.
- `OBJLoader2` has a new dependency `LoaderSupport` (see example).
- `WebVR.js` was rewritten. Check out the corresponding examples to see the new API.
- Renamed `CatmullRomCurve3`'s `type` to `curveType`.
- Removed `createPointsGeometry()` from `CurvePath`. Check out this example to see how to create a geometry from a series of points.
- Removed `createSpacedPointsGeometry()` from `CurvePath`.
- Removed `createGeometry()` from `CurvePath`.
- Renamed `Path`'s `fromPoints()` to `setFromPoints()`.
- Removed `extractAllPoints()` from `Shape`.
- Renamed `Mirror` to `Reflector`.
- Renamed `lengthManhattan()` of `Vector2`, `Vector3` and `Vector4` to `manhattanLength()`.
- Renamed `distanceToManhattan()` of `Vector2` and `Vector3` to `manhattanDistanceTo()`.
- Renamed `AxisHelper` to `AxesHelper`.

r86 → r87

- `GLTF2Loader` has replaced `GLTFLoader`.
- The result of the `onLoad` callback of `PDBLoader` has changed. Please have a look at the corresponding example.
- `AssimpLoader` now uses `LoadingManager`.
- Removed `setPreferredShading()` from `ColladaLoader`.

r85 → r86

- Removed deprecated `Animation`, `AnimationHandler` and `KeyFrameAnimation`
- Swapped y in setViewport() and setScissor()

r84 → r85

- `MultiMaterial` has been removed. Use an Array instead.
- Removed `multiplyToArray()` from `Matrix4`.
- Removed deprecated `SceneLoader`.
- `BoxHelper update()` no longer has arguments. Use `.setFromObject()` to assign a different object to the helper.
- `BoxHelper` no longer supports objects of type `Box3`.
- `DecalGeometry` now produces a `BufferGeometry`.

r83 → r84

- Removed `applyToVector3Array()` from `Matrix3`.
- Removed `applyToVector3Array()` from `Matrix4`.
- Removed `Spline`. Use `CatmullRomCurve3` instead.
- Removed `SplineCurve3`. Use `CatmullRomCurve3` instead.
- Removed `applyProjection()` from `Vector3`. Use `applyMatrix4()` instead.
- Renamed `Vector2`'s `fromAttribute()` to `fromBufferAttribute()`.
- Renamed `Vector3`'s `fromAttribute()` to `fromBufferAttribute()`.

- Renamed `Vector4`'s `fromAttribute()` to `fromBufferAttribute()`.
- Renamed `BinaryTextureLoader` to `DataTextureLoader`.
- Changed `Matrix4`'s `makeFrustum()` to `makePerspective()`.

r82 → r83

- `STLLoader` now produces a `BufferGeometry`.
- `PDBLoader` now produces a `BufferGeometry`.
- `AssimpJSONLoader` now produces a `BufferGeometry`.
- Renamed `Matrix3`'s `applyToBuffer()` to `applyToBufferAttribute()`.
- Renamed `Matrix4`'s `applyToBuffer()` to `applyToBufferAttribute()`.
- `BoundingBoxHelper` has been removed. Use `BoxHelper` instead.
- Renamed `XHRLoader` to `FileLoader`.

r81 → r82

- `PLYLoader` now produces a `BufferGeometry`.
- The `taper` parameter in `TubeGeometry` has been removed.

r80 → r81

- Renamed `Box2`'s `center()` to `getCenter()`.
- Renamed `Box2`'s `size()` to `getSize()`.
- Renamed `Box3`'s `center()` to `getCenter()`.
- Renamed `Box3`'s `size()` to `getSize()`.
- Renamed `Line3`'s `center()` to `getCenter()`.

r76 → r77

- `THREE.GridHelper`: `setColors()` removed, pass them in the constructor instead: `new THREE.GridHelper( size, step, color1, color2 )`.

r75 → r76

- `THREE.Audio` `.load` deprecated, use new `THREE.AudioLoader` instead.
- Uniforms no longer need a `.type` property.
- The uniform `boneGlobalMatrices` has been renamed to `boneMatrices`.

r74 → r75

- Changed `Vector3`'s `setFromMatrixColumn(index, m)` to `setFromMatrixColumn(m, index)`.
- Removed `WebGLRenderTarget`'s `shareDepthFrom`.

r73 → r74

- Renamed `enableScissorTest` to `setScissorTest`.
- Renamed `shadowBias` to `shadow.bias`.
- Renamed `shadowMapWidth` to `shadow.mapSize.width`.
- Renamed `shadowMapHeight` to `shadow.mapSize.height`.
- Renamed `shadowCameraNear` to `shadow.camera.near`.
- Renamed `shadowCameraFar` to `shadow.camera.far`.

- Renamed `shadowCameraFov` to `shadow.camera.fov`.
- Removed `shadowDarkness`. Add a `THREE.AmbientLight` to your scene instead.
- Removed `ClosedSplineCurve3`. Use `CatmullRomCurve3` with `closed` set to `true`.
- Removed `MeshPhongMaterial`'s `metal`.
- Renamed `Box2`'s `empty()` to `isEmpty()`.
- Renamed `Box3`'s `empty()` to `isEmpty()`.

r72 → r73

- Removed `morphColors` from `Geometry`.
- Removed `clampBottom` from `Math`.
- `FontUtils` and `TextGeometry` moved out of core.
- `shadowDarkness` default value is now `1`.

r71 → r72

- Renamed `PointCloud` to `Points`.
- Renamed `PointCloudMaterial` to `PointsMaterial`.
- Removed `computeTangents()` from `Geometry` and `BufferGeometry`.
- Moved all `shadowMap*` properties in `WebGLRenderer` to `shadowMap.*`.
- Removed `BufferGeometry`'s `drawcall.index`.
- `LineSegments( geometry, material )` should now be used instead of `Line( geometry, material, THREE.LinePieces )`.

r70 → r71

- Removed `ambient` from `Material`.
- Removed `recursive` parameter from `getObjectBy*()`.

r69 → r70

- Removed `sortParticles` from `PointCloud`.
- Removed `renderDepth` from `Object3D`.
- `UVMapping`, `CubeReflectionMapping`, `CubeRefractionMapping`, `SphericalReflectionMapping` and `SphericalRefractionMapping` are no longer functions.

r68 → r69

- `WebGLRenderer`'s `initMaterial` was made private.
- `ColladaLoader` now returns a `Scene` instead of an `Object3D`.

r67 → r68

- `Object3D`'s `position`, `rotation`, `quaternion` and `scale` properties are now immutable.
- `BufferGeometry`'s `addAttribute` method now takes a `BufferAttribute` instead of the various attribute types (e.g., `Int16Attribute`, `Float32Attribute`).

r66 → r67

- Removed `Face3`'s `centroid`.
- Removed `Geometry`'s `computeCentroids()`.

- Moved `GeometryUtils`'s `merge` to `Geometry`.

r65 → r66

- Renamed `CubeGeometry` to `BoxGeometry`.
- Removed `dynamic` property from `BufferGeometry`.

r64 → r65

- Removed `physicallyBasedShading` property from `WebGLRenderer`.

r62 → r63

- `WebGLRenderer` background to opaque (black) by default. Pass `{alpha=true}` when creating WebGLRenderer for previous default behaviour.

r61 → r62

- `Particle` removed. Use `Sprite` instead.
- `ParticleMaterial` removed. Use `ParticleSystemMaterial` or `SpriteMaterial`.

r59 → r60

- `Face4` removed. Use 2 `Face3` to emulate it.
- `OrbitControls`'s `zoomIn()` and `zoomOut()` renamed to `dollyIn()` and `dollyOut()`.

r58 → r59

- `Object3D.rotation` is now of type `THREE.Euler`.
- Removed `Object3D.useQuaternion`. The library now uses quaternions by default. However, there is some magic in place to keep `Object3D`'s `rotation` (`Euler`) working.
- Moved `Object3D.eulerOrder` to `Object3D.rotation.order`.
- Moved `Object3D.defaultEulerOrder` to `Euler.DefaultOrder`.
- Removed `setGeometry()` and `setMaterial()` from `Mesh`.
- Removed `Vector3.setEulerFromRotationMatrix()`, use `Euler.setFromRotationMatrix()` instead.
- Removed `Vector3.setEulerFromQuaternion()`, use `Euler.setFromQuaternion()` instead.

r57 → r58

- Removed `Matrix4`'s `translate()`, `rotateX()`, `rotateY()`, `rotateZ()`, `rotateByAxis()` and `crossVector()`.
- Removed `setClearColorHex()` from `CanvasRenderer` and `WebGLRenderer`. Use `setClearColor()` instead.
- Renamed `Matrix4`'s `extractPosition()` to `copyPosition()`.
- Renamed `Matrix4`'s `setRotationFrom*()` to `makeRotationFrom*()`.
- Renamed `Matrix4`'s `compose()` to `makeFromPositionQuaternionScale()`.
- Renamed `Object3D`'s `getChildByName()` to `getObjectByName()`.
- Removed `Object3D`'s `matrixRotationWorld` property.

r56 → r57

- For `BufferGeometry`
  - `geometry.verticesNeedUpdate` to `geometry.attributes.position.needsUpdate`
  - `geometry.elementsNeedUpdate` to `geometry.attributes.index.needsUpdate`
  - `geometry.normalsNeedUpdate` to `geometry.attributes.normal.needsUpdate`
  - `geometry.uvsNeedUpdate` to `geometry.attributes.uv.needsUpdate`
  - `geometry.colorsNeedUpdate` to `geometry.attributes.color.needsUpdate`
  - `geometry.tangentsNeedUpdate` to `geometry.attributes.tangent.needsUpdate`
  - `*` -> `geometry.attributes.custom.needsUpdate`
- Removed `Matrix4` 's `rotateAxis` . Use `Vector3.transformDirection( matrix )` instead.
- Removed `AsteriskGeometry` .
- Removed `Color` 's `setHSV` . Use `ColorConverter.setHSV( color, h, s, v )` instead.
- Renamed `JSONLoader` 's `createModel()` to `parse()` .

r55 → r56

- Removed `getPosition()` and `getColumn*()` from `Matrix4`
- `Color.setHSV()` and `Color.getHSV()` replaced by `.setHSL()` and `.getHSL()`
- Replaced `ColorUtils.adjustHSV()` with Color's `.offsetHSL()`
- Renamed `Box3/Line3/Plane/Ray/Sphere` 's `.transform()` to `applyMatrix4()`

r54 → r55

- `Matrix3.multiplyVector3()` changed to `Vector3.applyMatrix3()`
- `Matrix4.multiplyVector3()` changed to `Vector3.applyMatrix4()` and `Vector3.applyProjection()`
- `Matrix4.multiplyVector4()` changed to `Vector4.applyMatrix4()`
- `Quaternion.multiplyVector3()` changed to `Vector3.applyQuaternion()`
- Renamed `Color` methods:
  - `.lerpSelf()` to `.lerp()`
- Renamed `Vector2` , `Vector3` and `Vector4` methods:
  - `.add()` to `.addVectors()`
  - `.addSelf()` to `.add()`
  - `.sub()` to `.subVectors()`
  - `.subSelf()` to `.sub()`
  - `.cross()` to `.crossVectors()`
  - `.crossSelf()` to `.cross()`
  - `.minSelf()` to `.min()`
  - `.maxSelf()` to `.max()`
  - `.clampSelf()` to `.clamp()`
  - `.lerpSelf()` to `.lerp()`
- Renamed `Matrix4` methods:
  - `.multiply()` to `.multiplyMatrices()`
  - `.multiplySelf()` to `.multiply()`
- Renamed `Quaternion` methods:
  - `.multiply()` to `.multiplyQuaternions()`
  - `.multiplySelf()` to `.multiply()`

- Renamed `Frustum` methods:
  - `.contains()` to `.intersectsObject()`
- Moved `GeometryUtils.explode` to `ExplodeModifier`
- Moved `GeometryUtils.tessellate` to `TessellateModifier`
- Moved `ShaderUtils.lib` to `ShaderLib`
- `Matrix4.makeTranslation` and `Matrix4.makeScale` now take three scalars instead of `Vector3`.

r53 → r54

- `Sprite` material properties are now in `SpriteMaterial`, used like this `new THREE.Sprite( new THREE.SpriteMaterial( { color: 0xff0000, map: texture, alphaTest: 0.5 } ) )`.
- Renamed migrated sprite material properies: `Sprite.affectedByDistance` => `SpriteMaterial.sizeAttenuation` and `Sprite.mergeWith3D` => `SpriteMaterial.depthTest`
- Renamed `renderer.shadowMapCullFrontFaces` => `renderer.shadowMapCullFace` (default value `true` => `THREE.CullFaceFront`, other option `false` => `THREE.CullFaceBack`).
- Renamed `color.getContextStyle` to `color.getStyle`.
- Moved `Ray` casting methods to `Raycaster`.
- `Rectangle` replaced with `Box2`.
- `UV` replaced with `Vector2`. This means `.u` and `.v` are now `.x` and `.y`.
- `Matrix4.makeTranslation` and `Matrix4.makeScale` now take `Vector3` instead of three scalars.
- Moved `SubdivisionModifier` out of the build to `examples/js/modifiers`.
- Renamed and moved `Renderer.deallocateObject()` => `Geometry.dispose()` and `BufferGeometry.dispose()`.
- Renamed and moved `Renderer.deallocateRenderTarget()` => `WebGLRenderTarget.dispose()`.

r52 → r53

- `Sprite`'s size is no longer automatically based on the image size, use `sprite.scale.set( imageWidth, imageHeight, 1.0 )` to achieve the old behavior
- `SceneLoader` and scene format now use `widthSegments`, `heightSegments`, `depthSegments` instead of `segmentsWidth`, `segmentsHeight`, `segmentsDepth` for definitions of plane, cube and sphere geometries
- `SceneLoader` and scene format now use `material` property with single material id string instead of `materials` array for meshes
- `MeshPhongMaterial` now uses `perPixel = true` by default
- `WebGLRenderer` constructor doesn't use anymore `maxLights` parameter: shaders will be generated with the exact number of lights in the scene (it's now up to the application layer to make sure shaders compile on a particular system)
- `ColorUtils.rgbToHsv()` got moved into `Color.getHSV()`
- `Geometry` no longer has a `materials` property. `MeshFaceMaterials` usage is now like this: `new THREE.Mesh( geometry, new THREE.MeshFaceMaterial( [ material1, material2 ] ) )`. Meaning that `face.materialIndex` will map the array passed to `MeshFaceMaterials`.
- Loader callbacks which previously only had `geometry` parameter, are now also passed a second one: `materials`: `loader.load( 'file.js', function ( geometry, materials ) {} )`.
- `GeometryUtils.clone()` is now a method in `Geometry`.

r51 → r52

- `ShaderExtras` have been split in a different files. `CopyShader`, `FXAAShader`, `NormalShader`, etc
- Replaced `SceneUtils.traverseHierarchy` with `object.traverse`.
- Removed `SceneUtils.showHierarchy`. Use `object.traverse( function ( child ) { child.visible = false } )` instead.
- Moved `*Controls` to `examples/js/controls`.
- Split `SceneUtils.cloneObject` into `*.clone()`

r50 → r51

- `CameraHelper` API changes: helper is not anymore child of camera, instead it uses reference to camera world matrix
- texture uniform changes: texture units are now assigned automatically, texture object goes to `value` property instead of `texture` one `{ type: "t", value: 0, texture: map }` => `{ type: "t", value: map }`
- `normalScale` uniform in normal map shader is now `Vector2` (to be able to invert separately `x` and `y` to deal with different tangent bases)
- `CTMLoader.load` and `CTMLoader.loadParts` now use single parameter object for `useWorker` and `useBuffers`: `loader.load( url, callback, useWorker, useBuffers )` => `loader.load( url, callback, { useWorker: true, useBuffers: true } )`
- `CTMLoader` now creates `BufferGeometry` by default, set `useBuffers` parameter to `false` if you need `Geometry`
- type for non-procedural geometries changed in the scene format: `ascii_mesh` => `ascii`, `bin_mesh` => `binary`, `embedded_mesh` => `embedded`
- `UTF8Loader` (and compressor) were changed to a newer version, supporting more recent version of UTF8 format (`r100+`); loader doesn't create anymore geometries but instead it returns hierarchy with potentially multiple meshes created per each material (or by splitting large model)

r49 → r50

- `Vector3`'s `.getRotationFromMatrix( matrix, scale )` to `Vector3`'s `.setEulerFromRotationMatrix( matrix, order )`.
- `Vector3`'s `.getEulerXYZFromQuaternion( quaternion )` and `.getEulerYZXFromQuaternion( quaternion )` to `.setEulerFromQuaternion( quaternion, order )`.
- `DOMRenderer` and `SVGRenderer` no longer included in common build.
- texture coordinates aren't anymore flipped in the geometries, instead textures have `flipY` property (true by default); all models need to be re-exported / reconverted (tools have been updated as well).
  workaround: `uv.v = 1 - uv.v;`
- `PlaneGeometry` changed back to vertical orientation (facing screen instead of laying on the ground).
  workaround: `mesh.rotation.x = - Math.PI / 2;` or `geometry.applyMatrix( new THREE.Matrix4().makeRotationX( - Math.PI / 2 ) );`
- `doubleSided` / `flipSided` properties moved from `Object3D` to `Material`'s `side` property (`THREE.FrontSide`, `THREE.BackSide` and `THREE.DoubleSide`).
- `objectMatrix` in shaders was renamed to `modelMatrix`.
- JIT caching removed from `Animation`.
- `geometry.dynamic` is now `true` by default.

- `Three.js` build renamed to `three.min.js`.

r48 → r49

- changed `PlaneGeometry` from vertical to horizontal orientation.
- renamed `__dirtyXXX` attribute properties to `xxxNeedUpdate`.
- removed `Vertex` class, use `Vector3` instead.

r47 → r48

- Removed `intersectScene` from `Ray`. Use `intersectObjects( array )` instead.

r45 → r46

- `loader.load( { model: url, callback: callback } )` to `loader.load( url, callback )`.