
Blend4Web. User Manual

Release v16.05

Triumph LLC

May 31, 2016

Contents

1	Overview	1
1.1	What's Blend4Web	1
1.2	"About Engines"	1
1.3	Graphics Engine, Game Engine	1
1.4	What's WebGL	2
1.5	WebGL Browsers Support	2
1.6	Advantages of WebGL	3
1.7	What's Blender	3
1.8	3D Modeling	3
1.9	Browser Technologies	4
1.10	Interactive Graphics	5
1.11	Video Cards and Drivers	5
2	Engine Features	6
3	Installing and Updating	13
3.1	Installation	13
3.2	Updating	16
4	Workflow	18
4.1	Authoring Scenes	18
4.2	Exporting Scenes	19
4.3	Displaying Scenes in the Viewer	20
4.4	Application Development	22
5	Project Manager	23
5.1	Project Manager	23
5.2	Project Creation Wizard	25
5.3	Project Information	26
5.4	Project Import	28
5.5	Project Export	28
6	Scene Viewer	30
6.1	Navigation	30
6.2	The Side Panel	30

6.3	Indicators	33
7	Web Player	34
7.1	Usage	36
7.2	Navigation	36
7.3	Control Panel	36
7.4	Attributes	37
7.5	Scene Name as Title	38
7.6	Scene Errors	38
8	Addon	40
8.1	Local Development Server	40
8.2	Running Viewer and Demos	43
8.3	JSON	44
8.4	HTML	45
9	Scene Settings	63
9.1	Render Panel	65
9.2	Scene Panel	72
9.3	World Panel	79
10	Objects	83
10.1	Types	83
10.2	Static and Dynamic Objects	84
10.3	Settings	85
10.4	Anchor Settings	90
10.5	Object Transform API	92
10.6	Get Object API	93
10.7	Object Selection	94
10.8	Copying Objects (Instancing)	95
10.9	Quaternions	96
10.10	Differences Between Coordinate Systems of Blender and Blend4Web	98
10.11	Moving via TSR Vectors	99
11	Meshes	100
11.1	Static and Dynamic Meshes	100
11.2	Settings	101
11.3	Morphing	102
12	Normal Editor	104
12.1	Main Features of the Normal Editor	107
12.2	Interface of the Vertex Normal Editor	107
12.3	Activate	107
12.4	Show Normals	108
12.5	Rotation	108
12.6	Split Normals	109
12.7	Average Split	110
12.8	Restore	111
12.9	Tree, Foliage and Face	112

12.10 Copy/Paste	114
12.11 Copy From Mesh	115
13 Camera	118
13.1 Move Styles and General Settings	118
13.2 Limiting the Camera Movement	120
13.3 Viewport Alignment	128
13.4 Camera Controls API	129
14 Materials	136
14.1 Lighting Parameters	137
14.2 Transparency	138
14.3 Reflection	140
14.4 Rendering Properties	142
14.5 Viewport Properties	143
14.6 Engine Specific Parameters	144
14.7 Halo Materials	144
15 Node Materials	146
15.1 Standard Nodes	147
15.2 Engine Specific Nodes	148
16 Logic Nodes	161
16.1 Basics	161
16.2 Control Flow	164
16.3 Animation	170
16.4 Camera	173
16.5 Object	174
16.6 Operations	182
16.7 Sound	186
16.8 Network	187
16.9 Debug	191
16.10 Deprecated	192
16.11 Layout	193
16.12 Debugging	194
17 Lighting, Shadows and Background	195
17.1 Shading Types	195
17.2 Lighting with Light Sources	197
17.3 Environment Lighting (Ambient)	198
17.4 Shadows	200
17.5 Background	207
18 Postprocessing Effects	210
18.1 Motion Blur	210
18.2 Depth of Field	211
18.3 Screen-Space Ambient Occlusion	213
18.4 God Rays	214
18.5 Bloom	215

18.6	Outlining	216
18.7	Glow	217
18.8	Anti-Aliasing	218
19	Stereo Rendering	220
19.1	Activation	221
19.2	Additional Settings	222
20	Textures	223
20.1	Texture Types	225
20.2	Generic Settings	225
20.3	Diffuse Map	226
20.4	Specular Map	227
20.5	Normal Map	227
20.6	Height Map. Parallax Mapping	228
20.7	Stencil Map	229
20.8	Video Textures	231
20.9	Environment Map	233
20.10	Mirror Map	235
20.11	Skydome	236
20.12	Special Texture Types	238
21	Particle System. Fluids	242
21.1	Usage	243
21.2	Setup	244
21.3	Textures in Particle Systems	249
22	Particle System. Instancing	252
22.1	Particle System Setup	254
22.2	Grass	259
22.3	Tree Leaves	260
23	Animation	264
23.1	Animation Control	265
23.2	Object Animation	265
23.3	Skinning and Skeletal Animation	266
23.4	Vertex Animation	268
23.5	Default Animation	270
23.6	Non-Linear Animation	270
23.7	Audio Source Parametrization	272
23.8	Animation of Value and RGB Nodes	272
24	Outdoor Effects	275
24.1	Water	276
24.2	Atmosphere	284
24.3	Wind	287
25	Color Management	290
25.1	Gamma Overview	290

25.2	Human Vision and Monitors	291
25.3	Gamma Formula	291
25.4	Gamma in Node Materials	292
25.5	Alpha Compositing	293
25.6	Color Correction	294
26	Audio	296
26.1	Audio Source Settings	297
26.2	Processing and Decoding	298
27	Physics	300
27.1	Preparing for Use	300
27.2	Static Physics Type	301
27.3	Dynamic Physics Type	302
27.4	Constraints	304
27.5	Wheeled Vehicles	306
27.6	Floating Objects	308
27.7	Floating Vehicles aka Watercrafts	309
27.8	Use in Applications	310
28	For Application Developers	311
28.1	Project Management	312
28.2	Advanced Project Management	313
28.3	Application Programming	319
28.4	Resource Conversion	322
28.5	Code Examples	326
28.6	Event-Driven Model	326
28.7	SDK File Structure	328
28.8	Loading Local Resources	330
28.9	Quality Profiles	331
28.10	Non-FullScreen Web Apps	332
29	For Engine Developers	335
29.1	Coding Style	335
29.2	Building the Engine	337
29.3	Building the Addon	337
29.4	Dependencies	338
29.5	Naming Functions and Variables	338
29.6	Debugging	339
29.7	Shader Compilation	339
29.8	Updating Add-on Translations	345
30	Team Work. Using Git	346
30.1	Overview	346
30.2	Typical Workflow	347
30.3	Individual Settings	347
30.4	Checking the Status	348
30.5	Before the Commit	348
30.6	Preparing for Commit	349

30.7 Commit	351
30.8 Syncing Between Repositories	351
30.9 Resolving Conflicts	353
30.10 Tags	356
30.11 Other Useful Commands	356
31 Problems and Solutions	358
31.1 WebGL Support	358
31.2 Problems Upon Startup	359
31.3 WebGL Failed to Initialize	360
31.4 More In-Depth Troubleshooting	365
31.5 Known Issues	365
32 Release Notes	367
32.1 v16.05	367
32.2 v16.04	370
32.3 v16.03	372
32.4 v16.02	375
32.5 v16.01	379
32.6 v15.12	382
32.7 v15.11	385
32.8 v15.10	388
32.9 v15.09	392
32.10 v15.08	396
32.11 v15.07	399
32.12 v15.06	402
32.13 v15.05	407
32.14 v15.04	410
32.15 v15.03	414
32.16 v15.02	417
32.17 v15.01	420
32.18 v14.12	424
32.19 v14.11	426
32.20 v14.10	427
32.21 v14.09	429
Index	433

Overview

What's Blend4Web

Blend4Web is a web-oriented 3D engine - a software framework for authoring and interactive rendering of three-dimensional graphics and audio in browsers.

The platform is intended for visualizations, presentations, online-shops, games and other rich internet applications.

The Blend4Web framework is integrated tightly with Blender - a 3D modeling and animation tool (hence the name). The content is rendered by means of WebGL and other browser technologies, without the use of plugins.

Technically Blend4Web is a library for web pages, a Blender addon and some tools for debugging and optimization.

The Blend4Web 3D engine has been developed by Triumph LLC employees since 2010. The engine was first released on March 28 2014.

“About Engines”

An engine is a separate part of software code which is used by external applications for implementing the required functionality.

Engine examples are: site engine, blog engine, online shop engine, wiki engine, search engine, game engine etc. The economical reason for the existence of software engines is multiple usage of the same functionality. For example developers may create relatively cheap online shops or games using one or another engine.

Graphics Engine, Game Engine

A graphics engine performs special functions in displaying graphics. It is an intermediary between:

- high-level application part (game logic, business logic) and
- low-level system part (for example, the graphics library [WebGL](#) and underlying [drivers](#)).

A graphics engine may be combined with the sound system, the physics engine, the artificial intelligence system, the networking system and the scene and logic editors producing a three-dimensional engine - an integrated environment for authoring 3D applications.

What's WebGL

WebGL (Web Graphics Library) is one of the modern browser technologies which allows authoring 3D graphics applications. In other words WebGL is “3D in a browser”.

WebGL Browsers Support

At the moment WebGL is supported in to a varying degree by all browsers.

Full Support

- Chrome
- Yandex Browser
- Firefox
- Opera
- Safari 8+

Experimental Support

- Internet Explorer 11
- Microsoft Edge

Mobile Platforms

- Android
- BlackBerry
- Firefox OS
- iOS 8
- Sailfish OS

- Tizen
- Ubuntu Touch

Advantages of WebGL

- works in browsers without installing additional software (plugins)
- crossplatform, intended for all desktop and embedded systems
- [open standard](#), does not require licensing fees
- supported by the leading participants of the IT market (Google, Apple, Microsoft, Nvidia, Samsung, Adobe and others)
- based on OpenGL which is familiar to developers
- can be integrated with other browser technologies

What's Blender

Blender is a popular piece of software for 3D modeling and animation and is free and open source. Models and scenes which are created in this software can be displayed, for example, by means of a [three-dimensional engine](#) on a web page.

3D Modeling

Authoring graphics resources requires trained specialists - 3D artists.

A typical workflow may include the following stages:

- choosing photos and/or creating concepts and sketches (views from the front - from the side - from the above) of the future model or scene
- modeling - a 3D model consisting of polygons is created
- UV mapping - the model is unwrapped for further overlaying of textures (flat images)
- texturing - textures are overlayed on the 3D model
- materials setup - materials are assigned for different parts of the model and tuned (for example, a wooden door with a metal handle)
- rigging - the controlling elements ("skeletal bones") are attached to the model to animate it
- animation - the model is set in motion to visualize actions for example - of characters
- export - can be performed on any stage to display the 3D model in its final form, for example, on a web page

In addition, realism improving techniques are often used in the process of creating 3D models which require additional stages:

- creating a high-poly model - a detailed version of the model is created
- “baking” of a normal map - details from the high-poly model are transferred to the main model in the form of a special texture (normal map)
- creating a specular map - different reflection color and ratio are assigned to different model parts
- baking environment maps - is performed to visualize the surrounding environment reflection on the model surface
- setting up the camera and the light sources on the scene
- physical simulation parameters setup - particles, cloth

The time required to author 3D models and animation depends on their complexity and required quality and may vary from 1-2 days (for example a game item) to 1-2 weeks (for example a detailed aircraft model) and even to several months (realistic characters with clothing, hair, face sets, with animation and figure parameters setup).

Browser Technologies

Browser is a program for viewing Internet content. At the dawn of Internet technologies the browser’s role was to view text pages with the inclusion of static images (“hyper-text”). Modern browsers are full-scale platforms for multimedia web applications.

Among the already implemented and promising browser features which are used in Blend4Web the following technologies can be noted:

- three-dimensional graphics, [WebGL](#)
- [Typed Array](#)
- [Timing control for script-based animations \(requestAnimationFrame\)](#)
- two-dimensional graphics, [HTML Canvas 2D Context](#)
- sound processing, [Web Audio API](#)
- binary data loading, [XMLHttpRequest Level 2](#)
- [Fullscreen](#)
- [Pointer Lock](#)
- multithreading, [Web Workers](#)
- [Device Orientation](#)

Other promising technologies:

- [Scalable Vector Graphics \(SVG\)](#)
- [safe file access, File API, File API: Directories and System](#)

- real-time communication between browsers, [WebRTC](#)
- persistent network connection, [The WebSocket API](#)
- [Gamepad](#)

Interactive Graphics

Applied to computer graphics the term “interactive” means that the user can interact with a constantly changing image. For example the user can change the view direction in a 3D scene, move the objects, trigger animation and carry out other actions normally associated with computer games.

Graphics interactivity is achieved by utilizing a frequent change of images, so the user action (for example a mouse movement or the pressing of a key) between frames leads to the image changing in the next frame. Images must replace each other so frequently that the human eye could not recognize them individually (at least 30 frames per second).

“Real-time graphics” or “real-time rendering” are also similar in meaning to the term.

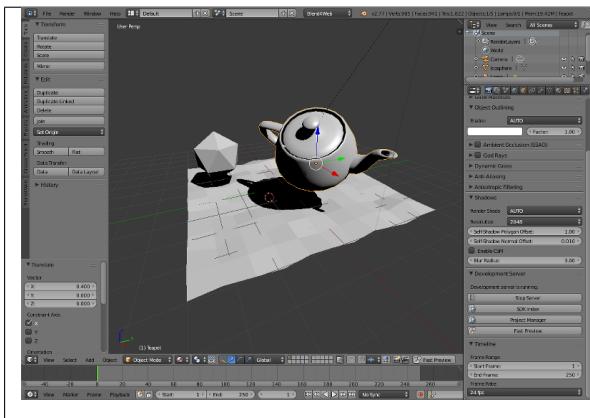
Video Cards and Drivers

Interactive graphics is provided by a special-purpose hardware part of modern computers so called graphics processor which can be implemented as a discrete device (video card) or as a part of the central processing unit.

Main graphics processors vendors for desktop computers are: - NVidia (GeForce, Quadro), AMD (Radeon), Intel (HD), for embedded devices - ARM (Mali), PowerVR (SGX), Nvidia (Tegra), Qualcomm (Adreno) (trade marks are specified in brackets).

Program access to graphics processor resources is carried out via an intermediate program called driver. It's important for the correct working of interactive graphics programs to have drivers of the latest version in the system. Drivers can be installed (or upgraded) from corresponding websites of graphics processors vendors. See detailed info in the section [WebGL Failed to Initialize](#).

Engine Features



BLENDER INTEGRATION

Creating interactive applications has never been easier! You can simply create a 3D scene in Blender and then use it in Blend4Web with the help of the following features:

- Fast and Convenient Scene Export
- One-Click Scene Preview
- Full Support for Blender's Environment and Material Settings
- Tools for Creating and Managing Projects

RENDERING

Blend4Web offers a great many features to make your creations look amazing.

Some of them include:

- 3D Scenes of Any Complexity
- Cascaded Shadow Mapping
- Dynamic Foliage
- Particle Systems
- Level of Detail
- Post-processing Effects
- VR Support





MATERIALS

Powerful and easy-to-use material creation system makes it possible to create any material you might need, from a simple cartoon-like to an almost photo-real one.

Some of the features:

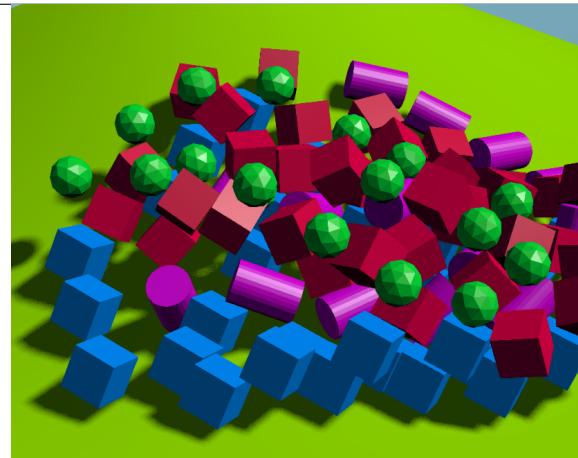
- Node Materials
- Parallax Offset Mapping
- Render-To-Texture
- Video Textures
- Full Support for Blender Material Nodes
- Refractions and Reflections

ANIMATION

Blend4web supports almost every animation technique available in Blender, including:

- Skeletal Animation
- Vertex Animation
- Object Animation
- Wind Bending
- Material Animation
- Animation Baking
- Flow Control Using API or Logic Editor





PHYSICS

If you are creating a physical simulation or simply want your scene to behave realistically, then Blend4Web's powerful physics system is at your service.

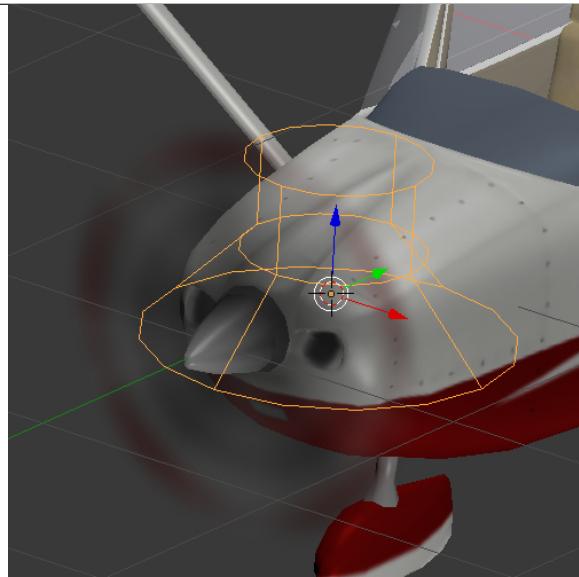
Its features include:

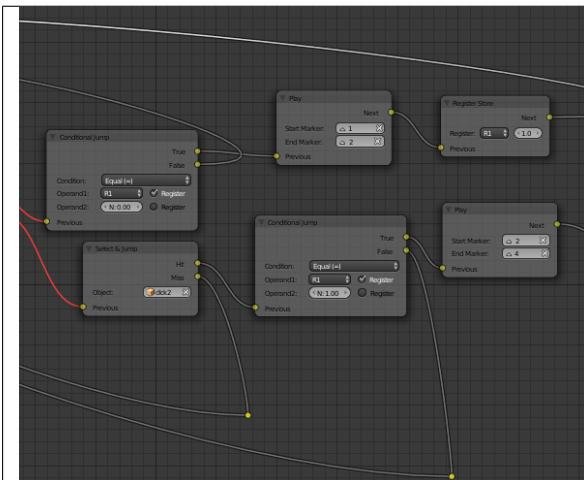
- Collisions
- Rigid Body Physics
- Joint System
- Ray Casting
- Floating Objects
- Vehicles

SOUND

Sound is important in making scenes and applications feel more responsive to user's actions and generally more alive. Here are some of the means that Blend4Web offers you to control how your creations will sound:

- Flexible Playback Control
- 3D Positioning
- Doppler Effect
- Dynamic Compressor
- Cross-fade Sound Animation
- Real-Time Mixing





LOGIC EDITOR

Using this visual editor, you can create logic node trees to control the behavior of your scenes and to add interactivity without writing single line of code!

Logic Node Editor features:

- Object Picking
- Animation Control
- Variables and Computations
- Server Data Exchange

SCRIPTING

Blend4Web uses JavaScript scenarios to control every aspect of an application. Some Blend4Web's JavaScript API features include:

- Module Structure
- Event-Driven Model
- GLSL Shaders
- Code Minification and Obfuscation

```
"use strict"

// register the application module
b4w.register("simple_app", function(application, require) {

    // import modules used by the app
    var m_app      = require("app");
    var m_data     = require("data");

    /**
     * export the method to initialize the app created at the
     */
    exports.init = function() {
        m_app.init({
            canvas_container_id: "main_canvas_container",
            callback: init_cb,
            show_fps: true,
            console_verbose: true,
            autoresize: true
        });
    }
});
```

Installing and Updating

Table of Contents

- Installing and Updating
 - Installation
 - * Installing Blender
 - * Unpacking the Archive
 - * Installing Blend4Web SDK
 - * Switching to Blend4Web Mode
 - Updating
 - * Updating the SDK
 - * Exporting and Viewing Scenes

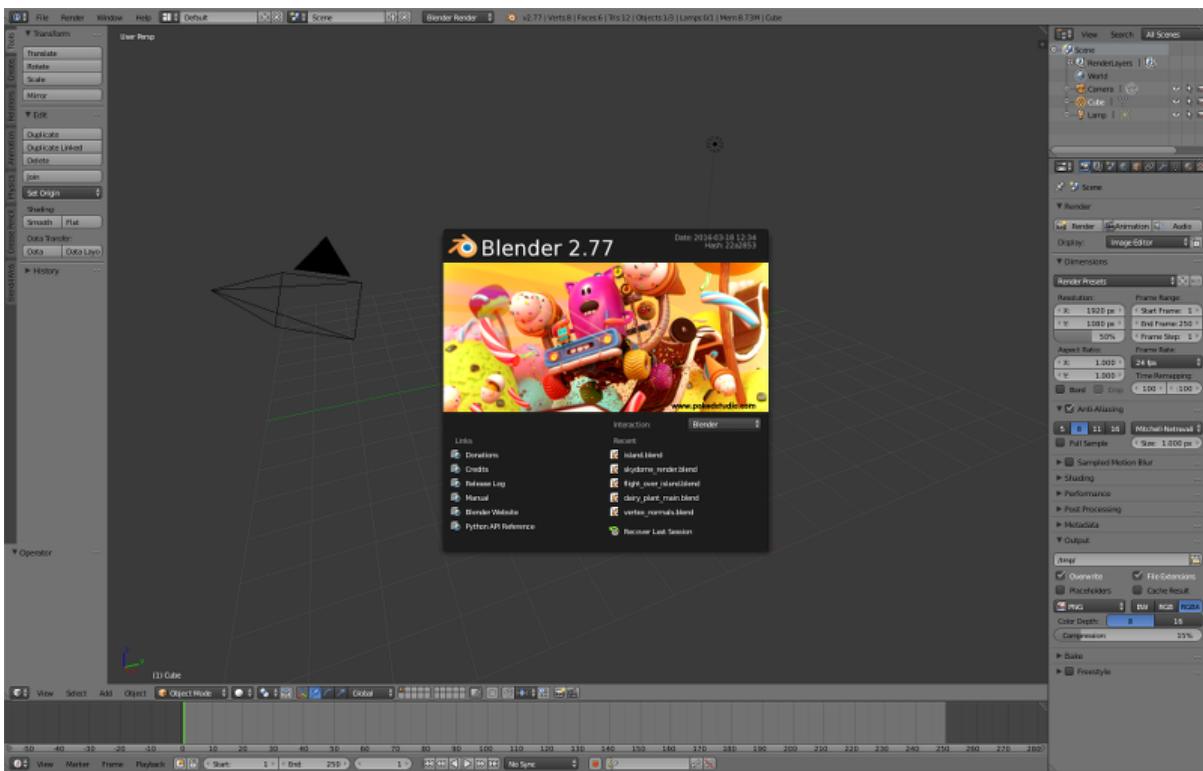
Installation

Installing Blender

Authoring 3D scenes is carried out directly in [Blender](#) which is open source software and is distributed free of charge.

Before installation, please download and install the compatible Blender version, according to this [table](#).

A current stable version of Blender should be used. It can be downloaded from the [official site](#).

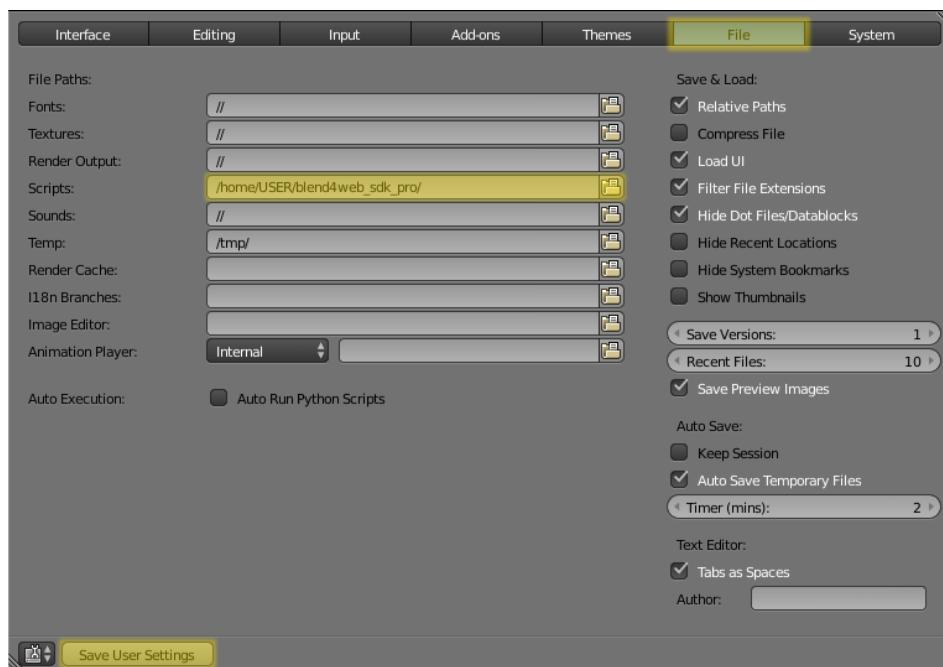


Unpacking the Archive

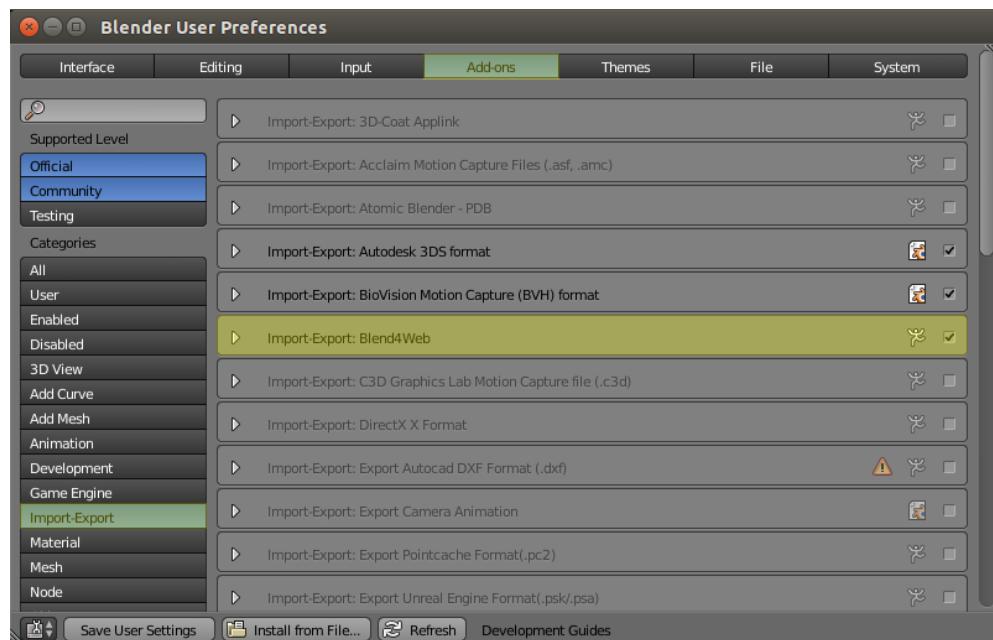
Stable versions of the distribution are available as an archive (blend4web_sdk_free_YY_MM.zip – free SDK, blend4web_sdk_pro_YY_MM.zip – commercial SDK). Simply unpack this archive somewhere.

Installing Blend4Web SDK

1. Run Blender.
2. Open the User Preferences panel **File > User Preferences...**
3. Open the **File** tab.
4. Set the path to the SDK directory in the **Scripts** field.
5. Click the **Save User Settings** button.



6. Restart Blender.
7. Once again, open the User Preferences panel File > User Preferences....
8. Open the Add-ons tab.
9. Enable the Blend4Web add-on.
10. Once again, click the Save User Settings button.



To verify it worked:

In the File > Export menu, the Blend4Web (.json) and Blend4Web (.html) options should appear.

Note: If you are planning to use only the normal editor, you might consider installing

the Blend4Web addon instead of the SDK.

The addon can only export scenes to [HTML format](#) and has limited functionality (for example, it doesn't have the [project manager](#), example scenes, user manual and so on), but still it has everything you might need to create a simple application.

It can be installed the same way as any other [Blender addon](#).

Switching to Blend4Web Mode

To reveal the engine settings, select Blend4Web from the upper panel menu:



Updating

Updating the SDK

Before updating, we recommend to backup all projects in development using the [project export](#) tools.

There are two methods to update the SDK.

Firstly, you can completely delete the SDK folder and, after that, extract the new version folder and set the path to it in the Blender settings, as if you were installing the SDK for the first time.

Secondly, you can simply overwrite the outdated SDK files with the new ones. This method is considered less “pure”, but in most cases it should not cause any problems.

After updating is complete, you can import all saved projects using the [project import](#) tools.

Exporting and Viewing Scenes

The created scenes can be exported in HTML format. To do this use the File > Export > Blend4Web (.html) or File > Export > Blend4Web (.json) menu option and choose the export filepath. Scene export is described thoroughly in [its own chapter](#). The resulting HTML file can be opened with any browser with WebGL support.

See also:

[WebGL Browsers Support](#)

Workflow

Table of Contents

- Workflow
 - Authoring Scenes
 - Exporting Scenes
 - Displaying Scenes in the Viewer
 - * Example
 - Application Development

Developing any product is a creative process with many participants who have different skills and experience. However, no matter how complex it is and what is the target it's always possible to separate the production stage in which the bulk of assets and source code is authored.

When using Blend4Web the workflow is the following:

1. Authoring 3D scenes in Blender.
2. Exporting scenes.
3. Running, tweaking and debugging scenes in the Viewer.
4. Creating the target application.

Authoring Scenes

Besides the usual stages such as modeling, texturing, animation etc a scene should be prepared for working in the engine.

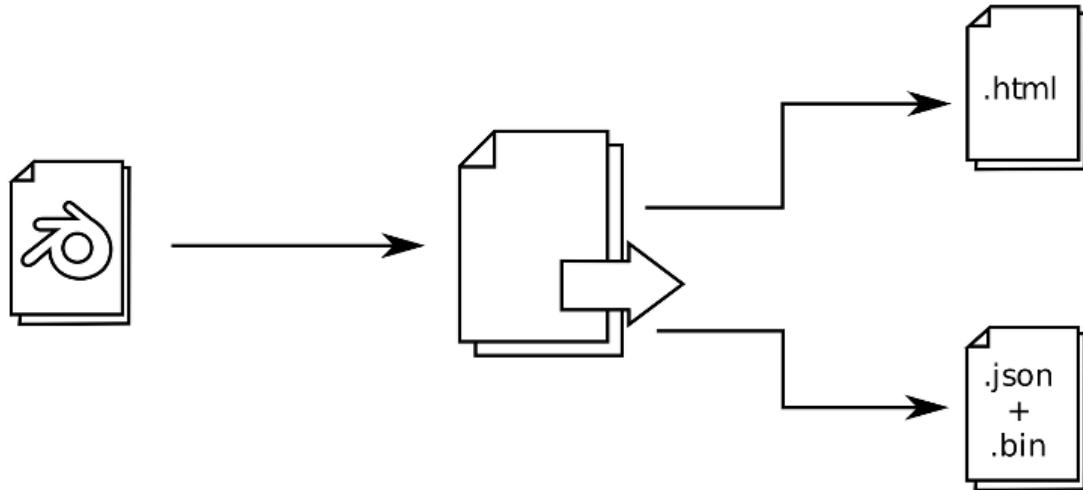
General recommendations:

1. We strongly recommend to save your blend files in the blender/project_name directory. Auxiliary files which are not intended for loading in your app (for example, references), should be located there as well.

2. Images and media files should be external and located in the deploy/assets/project_name directory.
3. Files from which you export should contain resources which are actually required in your application (objects, materials, textures, animation etc).
4. Scene components should have distinct names. They should not be named “Cube.001”, “Material” or “Armature”.
5. We recommend you to practice linking components from other files (libraries).

Exporting Scenes

The export procedure converts the scene from the Blender format to one of the formats used by Blend4Web.



There are two formats: JSON and HTML.

The JSON format is used for complex projects that require programming, while HTML is used for more simple, stand-alone applications. Format differences and export options are described in the [Export Formats](#) chapter.

To export the scene, select Blend4Web (.json) or Blend4Web (.html) option from the File > Export menu, or type b4w export in the operator search menu (available by pressing Spacebar).

Note: If the scene uses Blender features not yet supported by the Blend4Web engine, errors may occur during export. Export errors are listed in [this table](#).

Export options are described in detail in the [corresponding section](#).

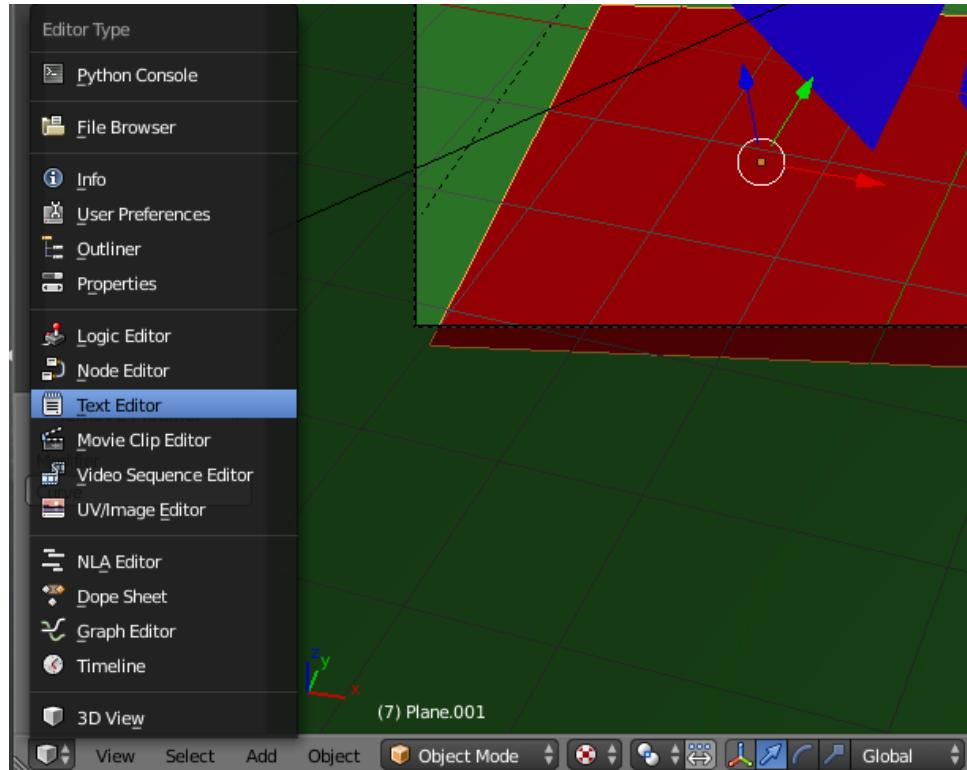
Displaying Scenes in the Viewer

When using the local development server it's possible to preview current scene using Fast Preview button, located on the Scene->Development Server panel in Blender. In this case the scene will be exported inside some temporary storage and loaded in the Viewer app.

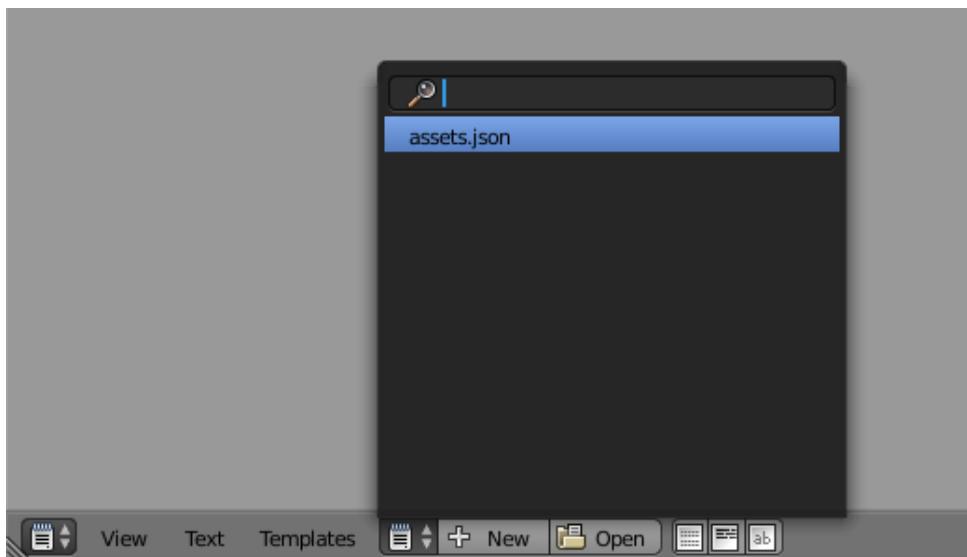
The other method is using [Run in Viewer](#) export option. In this case the scene will be displayed in the Viewer app immediately after the export.

For the long-term storage of the scene in the scene list of the viewer, it's required to manually add the entry to the apps_dev/viewer/assets.json text file. This file is opened automatically in Blender if you are using Blend4Web SDK.

Use the “Text Editor” to modify it:



Then select the assets.json file in the window to edit it:



To add a new scene you need to know the category in which it should be displayed. The category normally corresponds to the project name and to the name of the directory where the corresponding files are stored.

Example

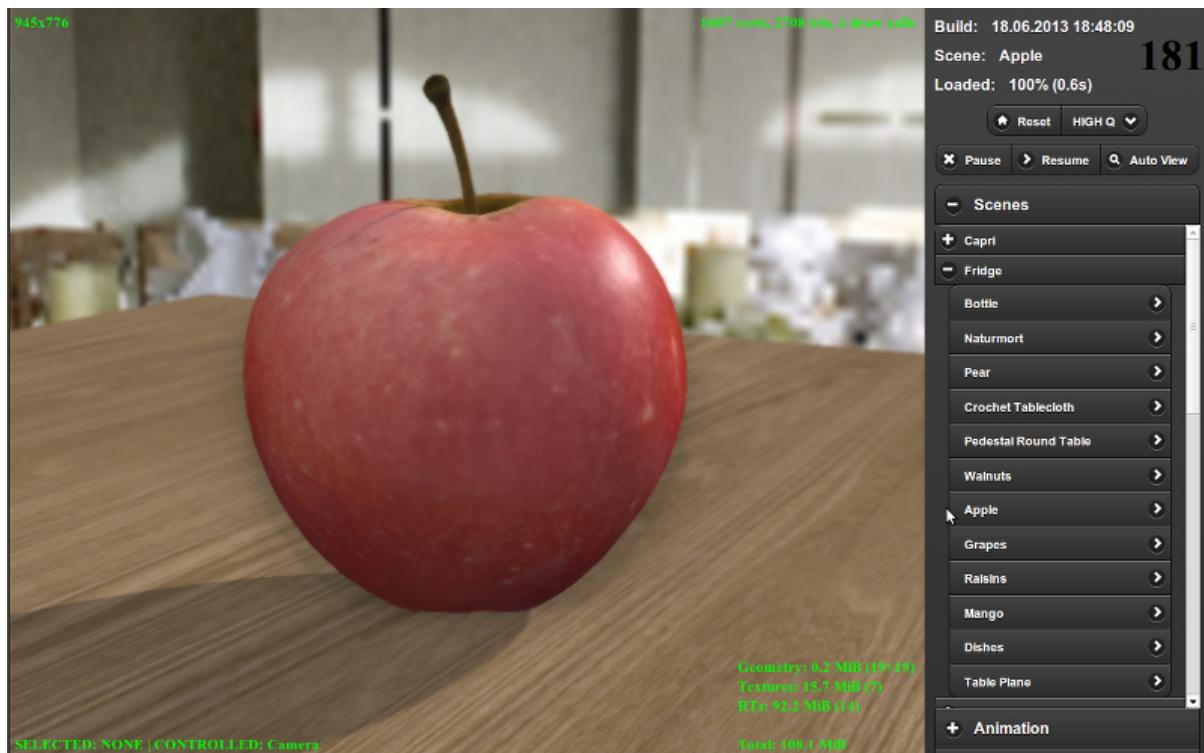
For example below you can see a part of assets.json. In this file there are two projects - “Capri” and “Fridge” each with corresponding scenes:

```
{
  name: "Capri",
  items: [
    {
      name: "Baken",
      load_file : "capri/props/baken/baken.json"
    },
    {
      name: "Terrain",
      load_file : "capri/landscape/terrain/terrain.json"
    }
  ],
  name: "Fridge",
  items: [
    {
      name: "Apple",
      load_file : "fridge/fruits/apple/apple.json"
    },
  ]
}
```

```
{
  name: "Mango",
  load_file : "fridge/fruits/mango/mango.json"
}
]
```

To add a new scene you can copy and paste a similar scene's description to the required category and then edit its name and path to the exported file.

A successfully added scene should appear in the scenes' list of the viewer in the required category



Application Development

At this stage an application is created. Logic for scene loading and user interaction is written using JavaScript. The application developer notes are given in the [corresponding](#) section.

Project Manager

Table of Contents

- Project Manager
 - Project Manager
 - Project Creation Wizard
 - Project Information
 - * Commands
 - * Project Parameters
 - Project Import
 - Project Export
 - * Commands
 - * Project Parameters

Since version 15.09, the SDK includes a project management feature, which makes it possible to:

- browse the full list and file structure of the projects;
- launch apps, run exported scenes in the Viewer, open source blend files in Blender;
- create and configure new apps, including those based on ready-made templates;
- build apps and pack them to be conveniently deployed on a remote server;
- convert app resources (textures, audio and video files) into alternative formats to ensure cross-browser and cross-platform performance;
- automatically re-export all scenes of an app, including export to JSON and HTML formats;
- remove projects.

Project Manager

The Project Manager app can be run from the Tools section of the SDK's index page. Upon launching, the app outputs a list of all current projects in the SDK.

The commands for project management are located at the top of the page.



Back to Index Returns to the Blend4Web SDK index page.

Create New Project Opens the project creation wizard.

Import Project(s) Opens the project import dialogue.

Export Project(s) Opens the project export page.

Hide Stock Projects Can be used to hide stock projects. If such projects are already hidden, this command is replaced with the Show Stock Projects command.

Help Opens the Help file.

Commands for managing a specific project is located at its right.



1. Compile project (not available for WebPlayer JSON and WebPlayer HTML projects).
2. Re-export blend files from the project.
3. Convert media resources.
4. Export and download a project archive.
5. Remove the project.

Note: All project paths are retrieved from its .b4w_project file.

Beside the project's name, a link to the project information page is located. The application type is also specified there. An application can have one of the following types:

Player The application can be played using the Web Player.

Dev Application for development.

Build Compiled application.

Project Creation Wizard

The tool for creating new projects is included in the Project Management app and executed by the [Create New Project] button on the main page of this app.

Project Manager

[← Back to Projects](#) [? Help](#)

1 Project Name (required) <small>for better compatibility, use only letters and underscore symbols</small>	<input type="text"/>
2 Project Title (optional) <small>choose a human-readable name</small>	<input type="text"/>
3 Project Author / Company (optional) <small>choose a human-readable name</small>	<input type="text"/>
4 Create Application Starter Files <small>create starter html/js/css files</small>	<input type="checkbox"/>
5 Create Scene Starter Files <small>create starter blend/json/bin files</small>	<input type="checkbox"/>
6 Copy Project Management Script <small>simplifies project management for users who prefer using command line tools</small>	<input type="checkbox"/>
7 Bundled Project <small>all project files will be placed into a single directory good for small projects such as tutorials etc</small>	<input type="checkbox"/>
8 Engine Binding Type <small>if unsure leave the default value see help for more info</small>	<input checked="" type="radio"/> External <input type="radio"/> Copy <input type="radio"/> WebPlayer JSON <input type="radio"/> Compile <input type="radio"/> Update <input type="radio"/> WebPlayer HTML
9 JavaScript Obfuscation Level <small>if unsure, leave the default value see help for more info</small>	<input checked="" type="radio"/> Simple <input type="radio"/> Advanced <input type="radio"/> Whitespace Only
✓ Create Project	

The [Back to Projects] button can be used to return to the Project Manager's main page, while the [Help] button can be used to access the [Help](#) file.

1. Project name is used to name project directories such as “apps_dev/project_name”, “deploy/assets/project_name”, “blender/project_name” and “deploy/apps/project_name”.
2. Project title as shown in the browser.
3. Project author’s name.
4. Add application templates. Standard application templates: html file, css file, js file will be added to the project directory “apps_dev/project_name”.
5. Add scene templates. Standard json file will be added to the “deploy/assets/project_name” directory; blend file will be added to the “blender/project_name” directory.

6. Copy project manager script. The project.py script will be copied to the project directory.
7. All project files will be located in the same directory. It is preferable to use this option in small projects, such as lessons and examples. Only “update” project type is available for this option.
8. Project’s type. Several options are available:
 - “External” - engine will be taken from the “deploy/apps/common/” directory. Only application files will be compiled;
 - “Copy” - engine is copied into the compiled application directory. Only application files are compiled;
 - “Compile” - engine sources are compiled with application scripts;
 - “Update” - engine inside the project directory will be replaced;
 - “Webplayer JSON” - json-file placed inside the project is run with the help of web-player inside SDK;
 - “Webplayer HTML” - project is packed into single html-file, containing all required resources.
9. Javascript optimization level.
 - “Simple” - variable names are replaced in the code;
 - “Advanced” - code optimization is performed;
 - “Whitespace Only” - only whitespaces are removed from the code.

Project Information

This page contains information regarding the selected project and can be viewed by the [info] link beside the project’s title.

Project Info	
← Back to Projects Help	
Project Name	flight
Project Title	The Island
Project Author / Company	Blend4Web
Project Icon	.b4w_icon.png
Applications	[detected automatically]
Engine Binding Type	External
Size	129 Mb
Project Path (Development Directory)	apps_dev/flight
Project Config	apps_dev/flight/b4w_project
Build Directory	deploy/apps/flight
Blend Directory(s)	blender/flight_over_island
Assets Directory(s)	deploy/assets/flight_over_island
URL Params	
JavaScript Obfuscation Level	advanced
JavaScript Compilation Ignore List	
CSS Compilation Ignore List	
Deployment Directory Assets Prefix	

[← Back to Projects](#)

Commands

[Back to Projects](#) Returns to the Project Manager's main page.

[Help](#) Opens the Help file.

Project Parameters

Project Name The name of the project.

Project Title Project title as shown in the browser.

Project Author / Company The name of the project's author or the title of the developer company.

Project Icon The icon of the project.

Application Application's main JSON file.

Engine Binding Type The type of the project.

Project Path (Development Directory) Project's directory.

Project Config Project's config file.

Build Directory Project build folder.

Blend Directory(s) Directories where project's blend files are located.

Assets Directory(s) Directories where project's media assets are located.

URL Params The list of [URL parameters](#) used to start the application.

JavaScript Obfuscation Level JavaScript optimization level.

JavaScript Compilation Ignore List The list of exceptions for the project's script compilation.

CSS Compilation Ignore List The list of exceptions for the projects style sheets compilation.

Deployment Directory Assets Prefix The scene resource folder in the deployed application.

Project Import

Tools for importing projects are available by the [Import Project(s)] link.

Clicking this link opens standard Open File dialog where you can select the project you need to import.

Project Export

Project exporting window can be accessed by the [Export Project(s)] link.

Export Project(s)

[← Back to Projects](#)
 [Hide Stock Projects](#)
 [Help](#)

Select	Name	Title	Author
<input type="checkbox"/>	boat_pf28	Azure Grotto	Blend4Web
<input type="checkbox"/>	capri	Capri	Blend4Web
<input type="checkbox"/>	capri_char	Capri Char	Blend4Web

<input type="checkbox"/>	webplayer	Web Player	Blend4Web
<input type="checkbox"/>	website	Website	Blend4Web

Archive Name: project
.zip

Commands

[Back to Projects](#) Returns to the Project Manager's main page.

[Hide Stock Projects](#) Hides stock projects. If such projects are already hidden, this command is replaced with the [Show Stock Projects] command.

[Help](#) Shows the Help page.

[Export Project\(s\)](#) Can be used to export selected projects.

Project Parameters

Select Shows if the project is selected for export.

Name The name of the project's directory.

Title Project's title.

Author Project author's name.

Archive Name The name of an archive to which exported projects are packed.

Scene Viewer

Table of Contents

- Scene Viewer
 - Navigation
 - The Side Panel
 - * Control elements list in top-to-bottom order
 - Indicators

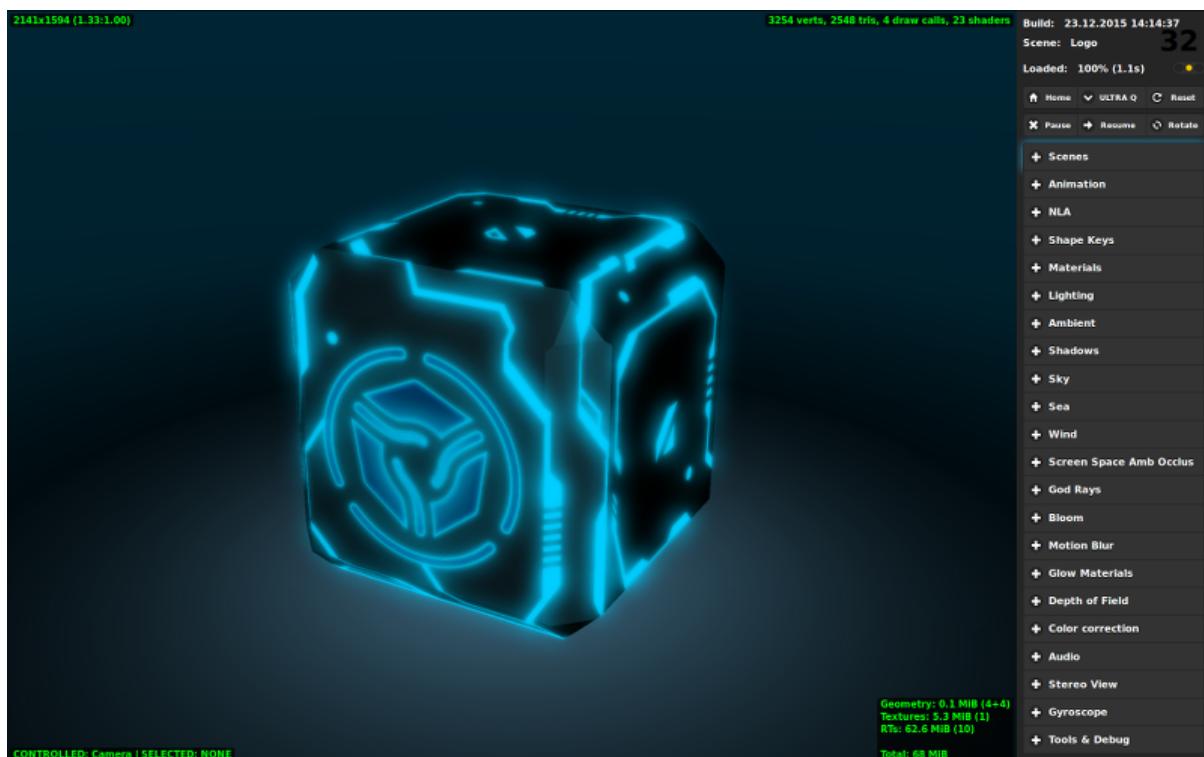
Running The Scenes Viewer.

Navigation

To control the camera hold down a mouse button and move the mouse. Control can also be performed using the W, A, S, D, R, F keys: forward, left, back, right, up, down. Arrows and numpad keys can be used as well. In the Target camera mode it's possible to focus on the selected object using the Z or .(dot) keys.

The Side Panel

The side panel consists of three areas: the information board, basic control buttons and the list of drop-down panels with additional control elements differentiated by functionality.



Control elements list in top-to-bottom order

Build The engine build date and time. In the developer version this shows the page load time.

Scene Loaded scene name from the assets.json file. Path to the file pops-up on mouse hover.

Loaded Loading progress and time.

Reset This button deletes the saved name of the last viewed scene and reloads the page back to display the default scene.

LOW Q - HIGH Q - ULTRA Q Drop-down menu for choosing the performance profile of the engine.

See also:

[Quality Profiles](#)

Pause Pause rendering.

Resume Resume rendering.

Auto View Activate the automatic scene switching mode; the delay between views is 1 second.

Scenes A double-level list of the categories and scenes from the assets.json file.

Animation Animation controls. When viewing animated models, you can select an object and switch its animation with a drop-down menu, switch cyclic animation mode, stop and resume animation, set the required frame (the animation should be stopped to do this).

NLA A tool to control Non-Linear Animation (if it is present in the scene). For this panel to work, NLA option should be enabled in the scene's settings.

Shape Keys Shape Keys control. When viewing models that has shape keys, you can select an object, one of its shape keys and the influence that the key has on the object.

Materials Material properties setup. A material can be selected using the drop-down menu. For now, only a limited number of properties is supported.

Lighting Direct lighting parameters setup. A light source can be selected using the drop-down menu. Changing color and intensity is supported. Daytime and sun lighting parameters can also be tweaked on this panel.

Ambient Ambient lighting parameters setup. Changing the colors and intensity of a hemispheric ambient model is supported.

Shadows Shadow parameters setup, including shadow cascades and shadow edges softening parameters.

Sky Dynamic sky parameters setup such as color, sun light scattering parameters etc.

Sea Water rendering parameters setup, including color transitions by depth and by shore distance, foam and subsurface scattering parameters, waves dynamics etc.

Wind Wind parameters setup, including direction and strength.

Screen Space Amb Occlus Ambient occlusion parameters setup.

God Rays Crepuscular rays effect parameters setup.

Bloom Bright light effect parameters setup.

Motion Blur Motion blur effect settings.

Glow Materials Glow Materials effect settings.

Depth of Field Depth of field effect parameters setup.

Color correction Color correction parameters setup, including brightness, contrast, exposure and saturation.

Audio There is a mixing mode switch on the panel. After it is enabled the mixer interface becomes visible (only for scenes with sound sources).

Stereo View There is a stereoscopic mode switch on the panel.

Gyroscope Gyroscope settings for mobile devices can be controlled from this panel.

Tools & Debug This panel contains a range of debugging tools, including the wireframe mode and the postprocessing stages viewer switches.

Indicators

Frames per second counter This is located in the top right corner. It displays the averaged and rounded value for the last 1.5 seconds.

Viewport dimensions This is located in the top left corner. It displays the viewport dimensions in pixels.

Selected object and controlled object This is located in the left bottom corner. It displays the names of selected and controlled objects. Object selection can be performed with the mouse. To control the object directly (normally for physics debugging) press the Q key and click on the object. The object movement is performed with the W, A, S, D keys. To exit the control mode press the Q key and click on an empty space. The indicator also displays the distance to the selected object in Blender units (meters equivalent).

Scene complexity indicator Is located in the top right corner of the rendering area. It displays the number of vertices, triangles and WebGL calls on the main rendering scene (i.e. shadow rendering calls are not included, for example).

Video memory indicator Is located in the bottom right corner of the rendering area. It displays the amount of video memory used by geometry, textures, render targets, and also the total memory usage.

Scene load errors indicator Is located under the FPS counter. Shows errors and warnings which occurred during scene load. Red light means errors, yellow - warnings and green means that the scene was loaded successfully.



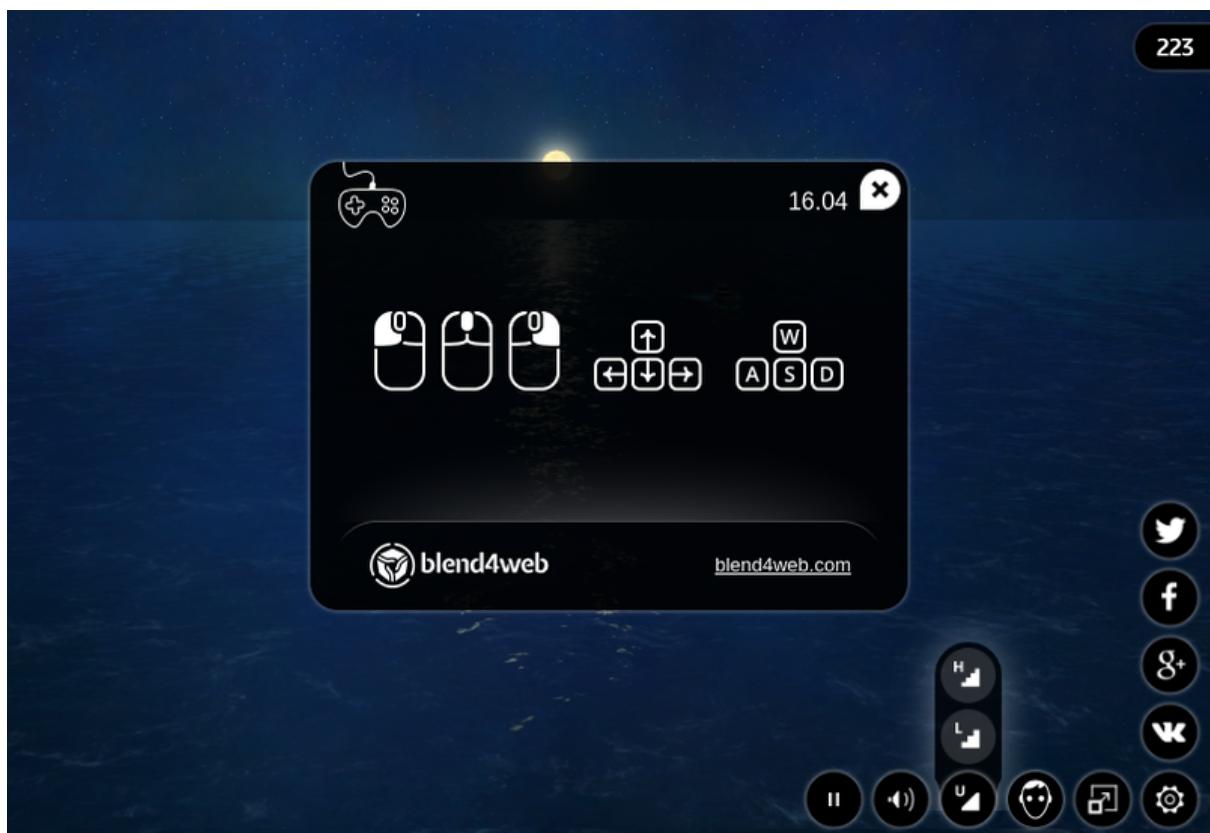
Web Player

Table of Contents

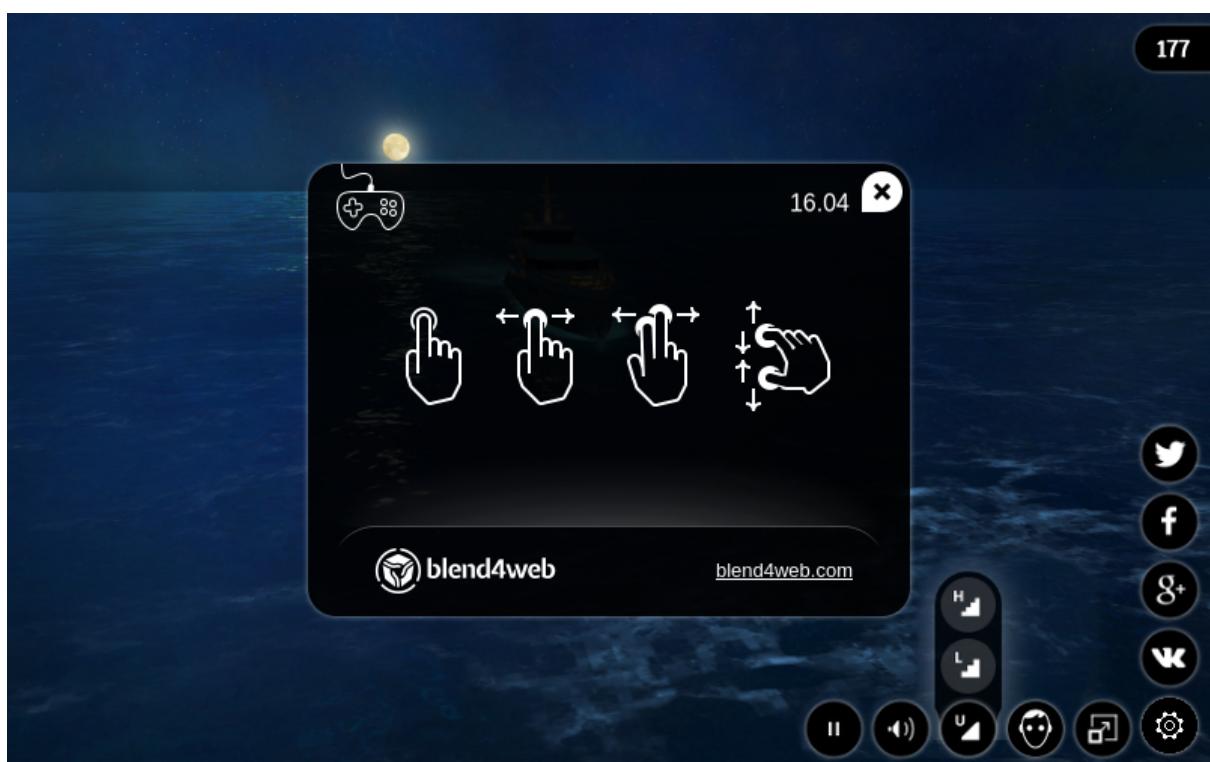
- Web Player
 - Usage
 - Navigation
 - Control Panel
 - Attributes
 - Scene Name as Title
 - Scene Errors

The web player is a special application for rendering models and scenes in a demonstration mode.

Desktop version:



Mobile version:



Usage

You can copy the directory containing the web player files, namely `deploy/apps/webplayer`, from the Blend4Web SDK distribution and deploy it on your website. You can place the exported scene files on your website and specify the path to them (absolute or relative) with the `load web player` parameter.

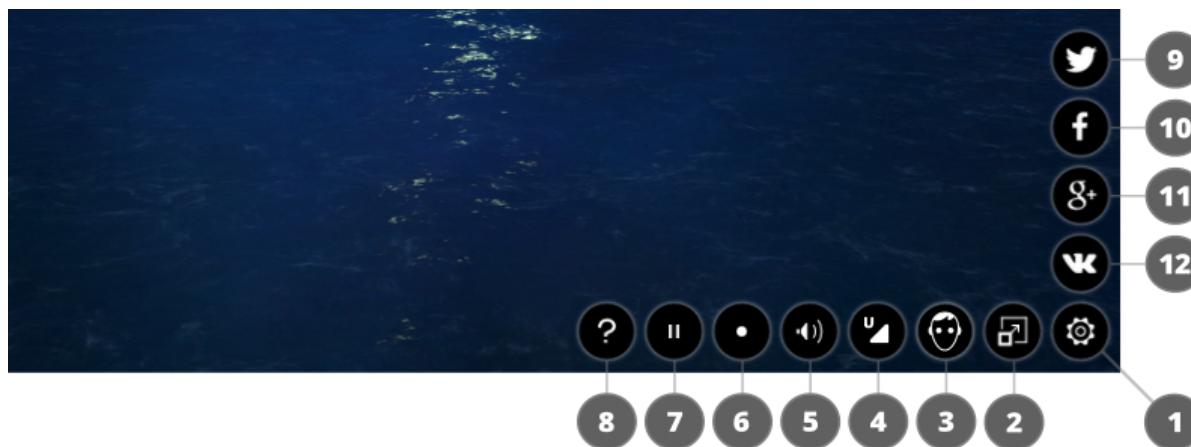
When you export into a single HTML file the web player interface is integrated automatically into it.

Navigation

The camera (in the Target and Eye modes) is controlled by the mouse with its button pressed or with the keys: W, A, S, D, R, F (forward, left, back, right, up, down). The numpad keys are also supported.

Control Panel

The web player's control panel is shown below.



1. show / hide control panel;
2. fullscreen mode on / off;
3. stereo rendering on / off;
4. set the scene quality;
5. sound on / off;
6. camera auto rotation mode on / off;

7. run / stop the engine;
8. open the help window;
9. tweet;
10. share via Facebook;
11. share via Google+;
12. share via VK.

Attributes

Web player accepts attributes from the browser address line:

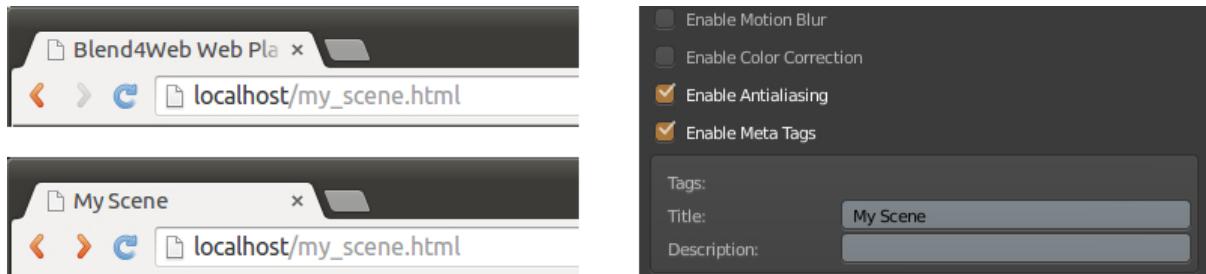


1. the special attribute load is used to load the scene, this attribute contains relative path to a JSON file.
2. in case of a WebGL error the optional fallback_image attribute is used to setup the background image instead of 3D content.
3. in case of a WebGL error the optional fallback_video attribute is used to setup the background video instead of 3D content. Can be used many times to add more video formats.
4. the optional show_fps attribute is used to display the FPS counter in the player's top right corner.
5. optional parameter autorotate is used to enable automatic camera rotation just after the scene loads.
6. the compressed_textures optional parameter is used to enable loading of minified and compressed textures (in DDS format).
7. optional alpha parameter is used to enable transparency for the rendering area.
8. the optional no_social attribute is used to disable social networks buttons on the control panel.

Note: If both fallback_image and fallback_video parameters are specified, the fallback_image parameter is used.

Scene Name as Title

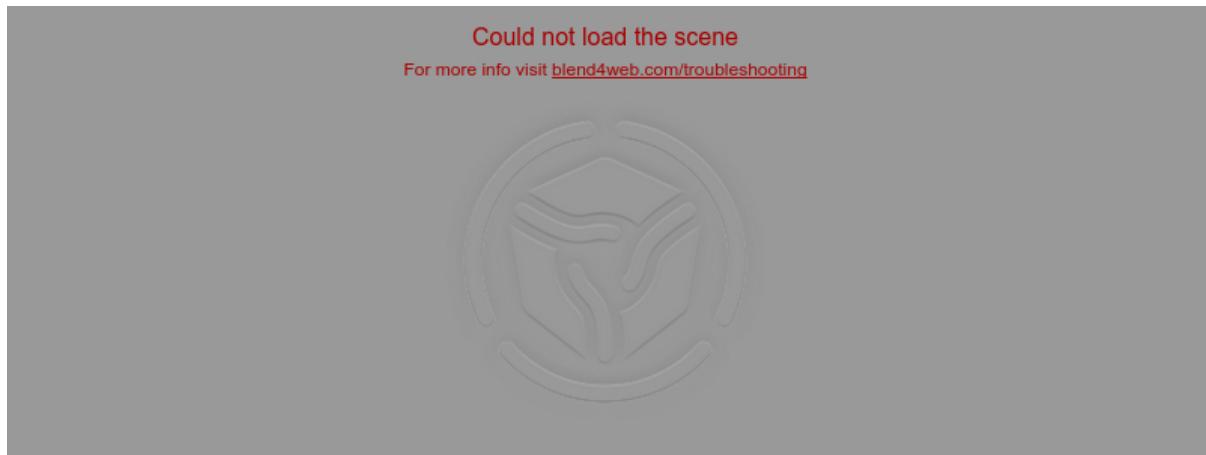
By default the Web Player has the Blend4Web Web Player title. Assigning the meta tag title on the scene in Blender you can change that value to something else.



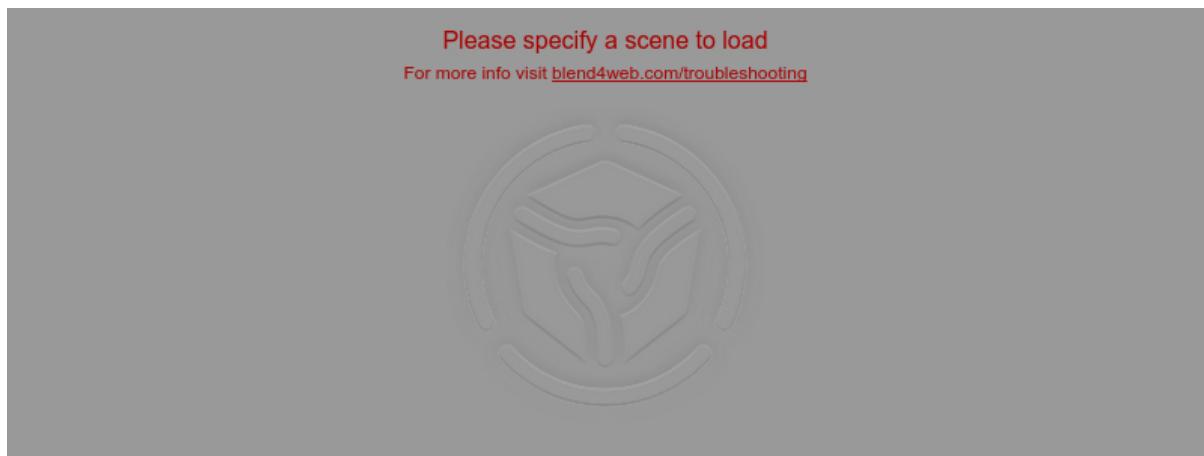
Scene Errors

If the player is used incorrectly it displays the corresponding errors.

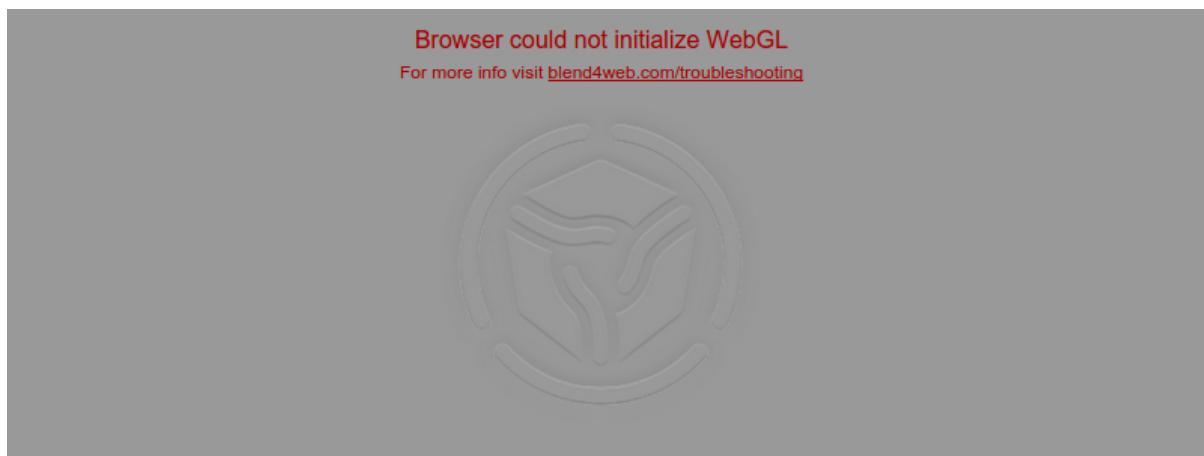
1. the load attribute specifies a wrong path to the JSON file or the file is corrupt;



2. the load attribute is not found or is void;



3. WebGL initialization error.



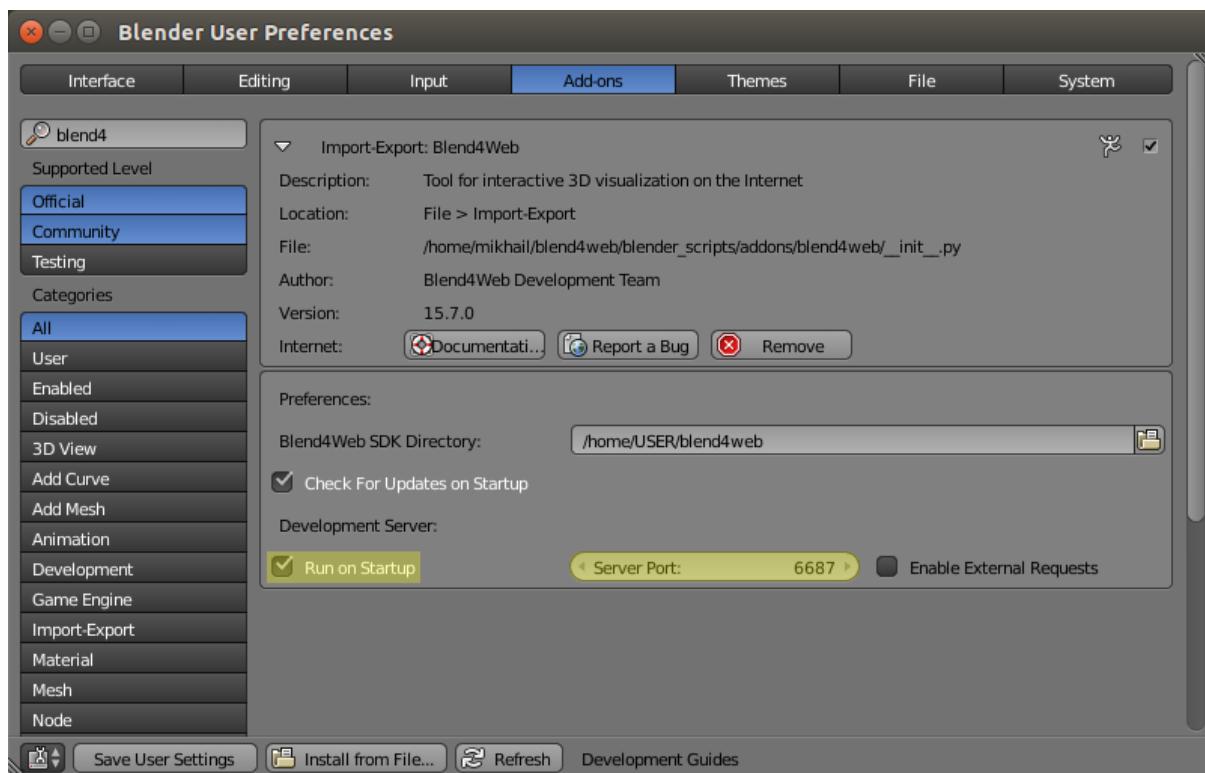
Addon

Table of Content

- Addon
 - Local Development Server
 - Running Viewer and Demos
 - * Export Formats
 - JSON
 - HTML
 - * Export Options
 - * Initialization Errors
 - * Compatibility Errors
 - * Critical Export Errors
 - * Non-Critical Export Errors
 - * Other Messages
 - * Add-on Translations

Local Development Server

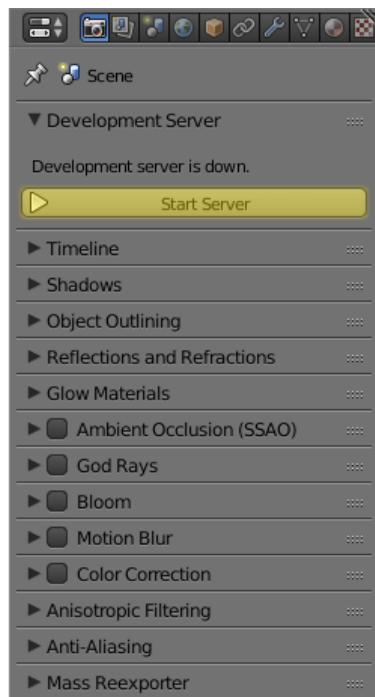
Settings for the local development server can be found in File > User Preferences... (hot keys Ctrl-Alt-U). Here you can change the port number to be used to run the development server (6687 by default), and also enable its launching upon Blender startup. To do this, enable Run on Startup option in the add-on settings.



After changing local development server settings in it required to restart Blender.

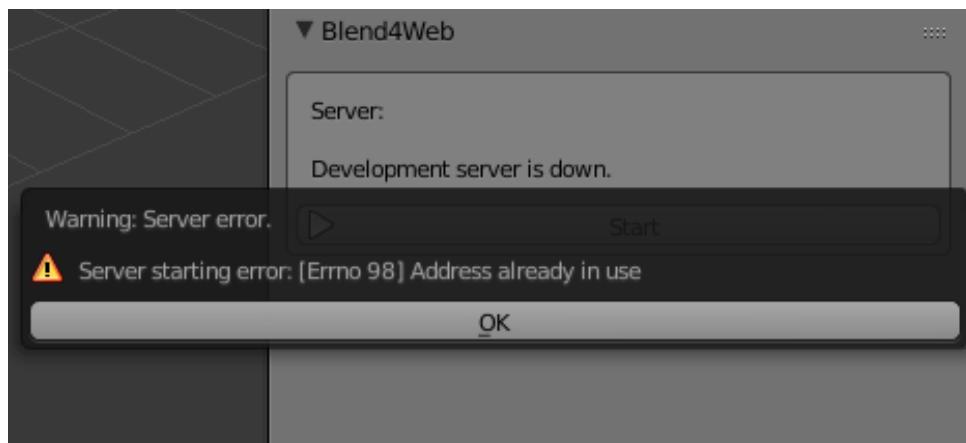
It is possible for the local server to process external requests. To do that enable the option Enable External Requests.

If you chose not to start the server automatically, you can always do it manually: go to the Render tab and press the Start Server button on the Development Server panel:



Note: If the path to the Blend4Web SDK is not specified, the local development server cannot be launched. In this case the corresponding message will be displayed instead of the Start Server button.

If the server is failed to run, an error message will be shown describing the reason:



This error can arise if the server port is already used by some other application.

Press the SDK Index button to open the index web page of the Blend4Web SDK in the browser. This page is available at <http://localhost:6687>.



As a result, the default browser for your operating system will be launched.

The Project Manager button can be used to open [project manager](#).

The Fast Preview button loads the scene that is currently open in Blender, into the viewer.

The server can be stopped by pressing the Stop Server button. It also stops when Blender is closed.

Running Viewer and Demos

The index page contains links for launching the scene viewer and the demo applications. A WebGL-capable browser is required to run these apps.

Blend4Web SDK Pro Free			
WebGL: available			
Development Server: available			
Tools			
Project Manager		Scene Viewer	
Code Snippets		WebGL Report	
Helpful Resources			
	<i>English</i>	<i>Русский</i>	<i>中文</i>
User Manual, HTML	read	читать	
User Manual, PDF	read	читать	-
API Reference	read		
Tutorials	go	перейти	
FAQ	go	перейти	
Community Support Forums	go	перейти	
Support Email	support@blend4web.com		

Note: If the SDK apps are not displayed correctly, or error messages are shown, follow the instructions in the [Problems Upon Startup](#) section.

Export Formats

After the scene is finished, you need to convert it into a format supported by the Blend4Web engine.

For now, two formats are supported: JSON and HTML.

JSON

Exporting the scene to this format creates a .json (JavaScript Object Notation) file that contains all exported data structures and links to external resources (images, sounds and

such), and also a .bin file that contains model data arrays in binary format.

If media resources are [packed into the .blend file](#), they will be unpacked during the export and placed to the hidden .../tmp/ folder in the project's directory (inside the SDK directory). It should be noted that the names of such files will be automatically changed, which can complicate working with them.

It is recommended to store all the resources in a dedicated folder inside the SDK directory, it can be called deploy/assets/project_name for example.

Note: In accordance with the security measures, the development server can only access the SDK folder. If media data is placed in another folder, the server won't be able to deploy it during the export (even if it is working correctly in Blender itself).

Paths to the external resources should be relative. If this is not the case, execute the File > External Data > Make All Paths Relative command, or else problems with opening the file on other computers may occur.

This is the main format for complex projects that include multiple scenes and require JavaScript programming. Project development is further described in the [corresponding section](#).

HTML

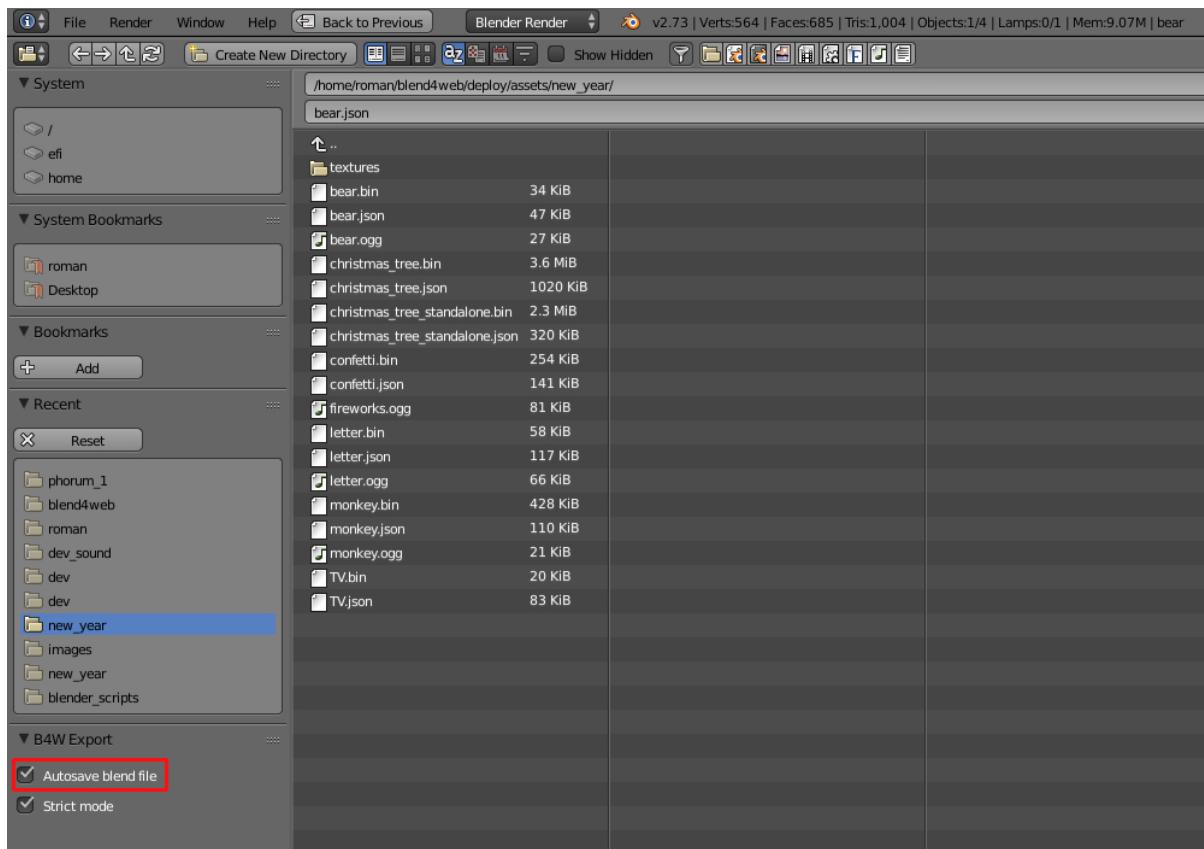
Exporting the scene to this format pack all scene resources into one file with the HTML extension. This HTML file contains not only the scene itself, but also textures, sounds, Blend4Web engine and standard [web player](#). A file like this can be executed on any computer and any mobile device that have a web browser with WebGL support.

You can't use HTML files for further development, but you also don't need any additional actions to run them. This format is useful for developing relatively simple applications of moderate size (?).

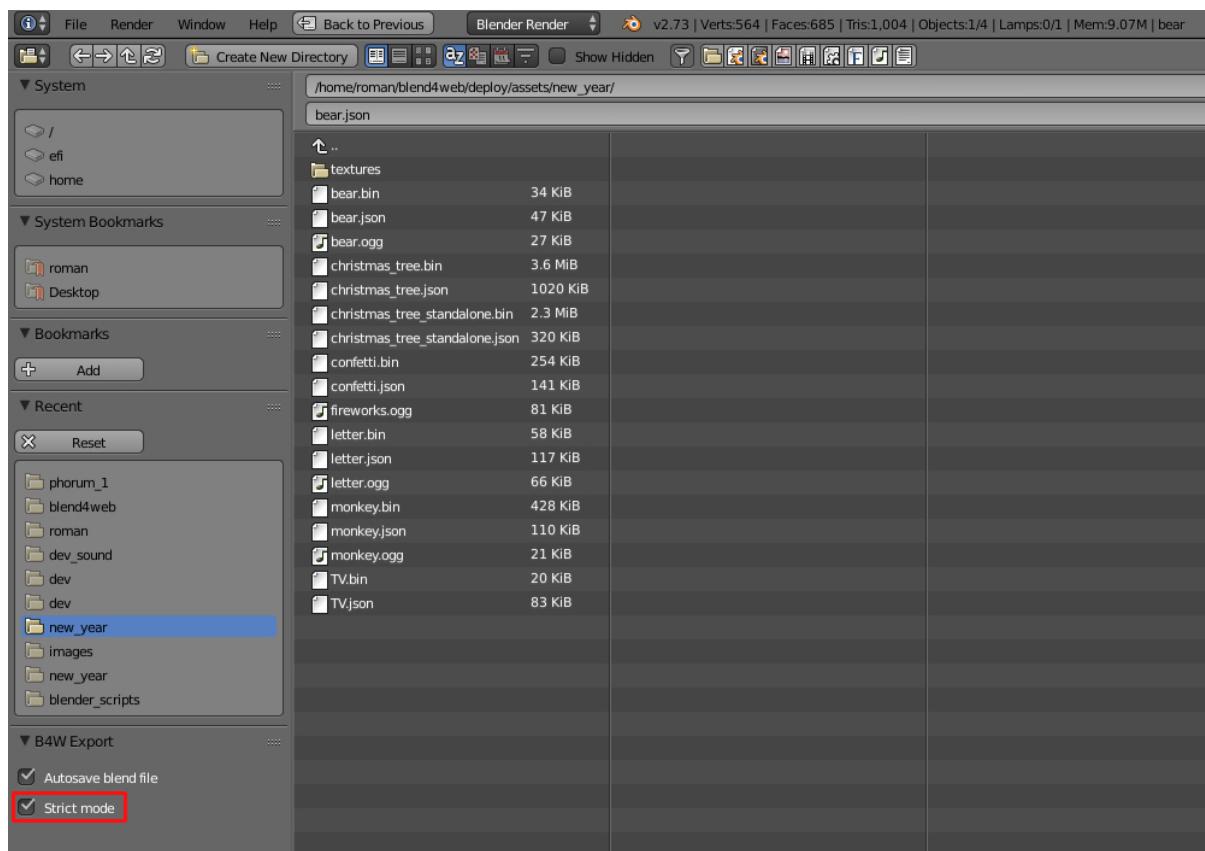
Export Options

Autosave blend File Autosaving the file from which export occurs. Enabled by default.

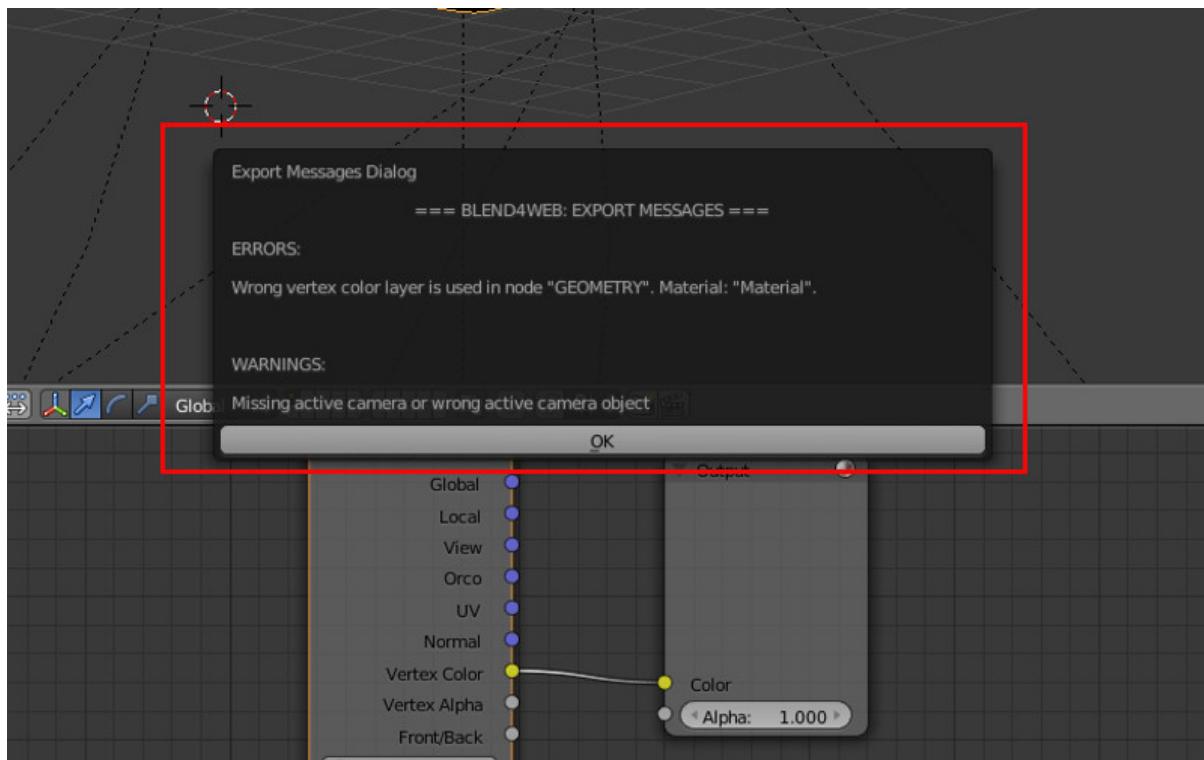
Autosaving is performed right after the export to guarantee conformity between the current blend file and the exported file contents. In addition, the relative path to the exported file is saved for convenience.



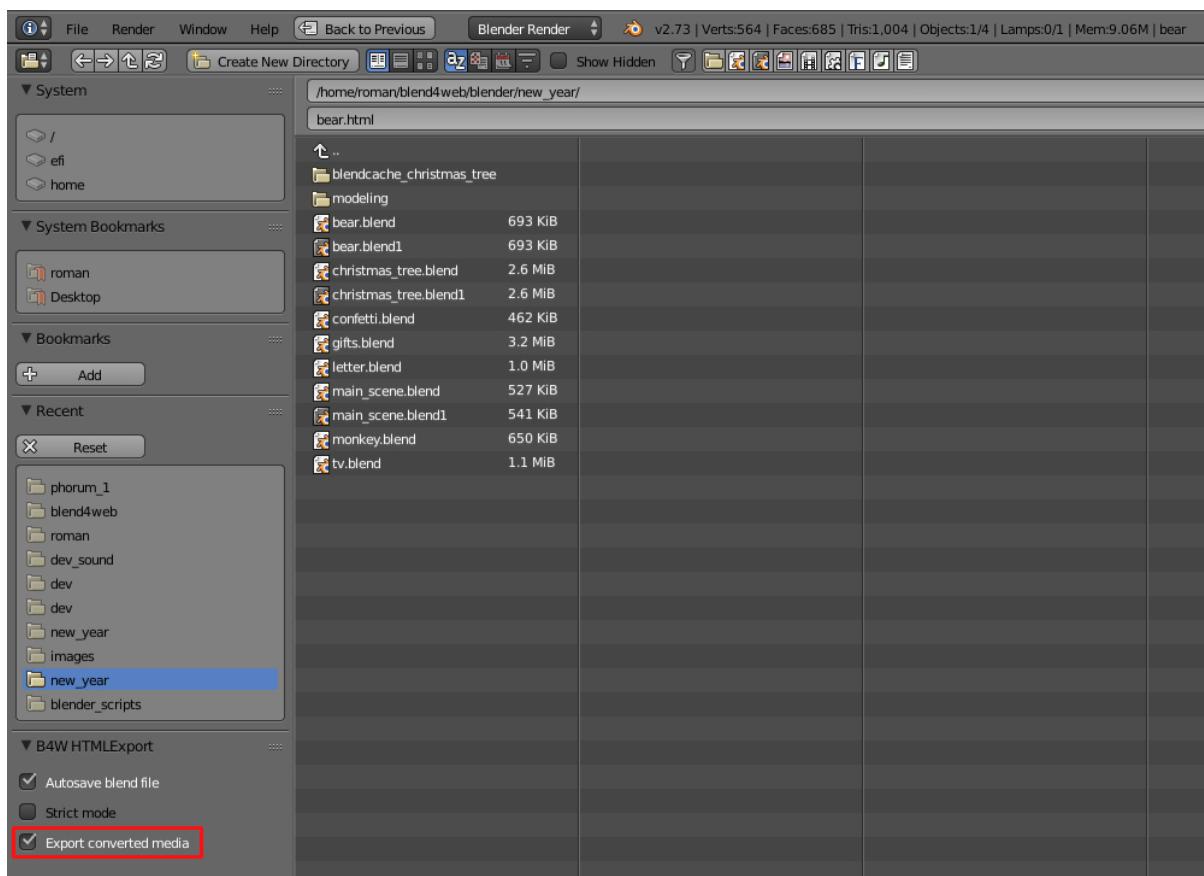
Strict Mode This mode prevents export if there are any errors or messages for users' attention. This mode is enabled with the Strict Mode setting in the export menu:



If there are any non-critical errors or messages for users' attention, a dialog window will be show like this:



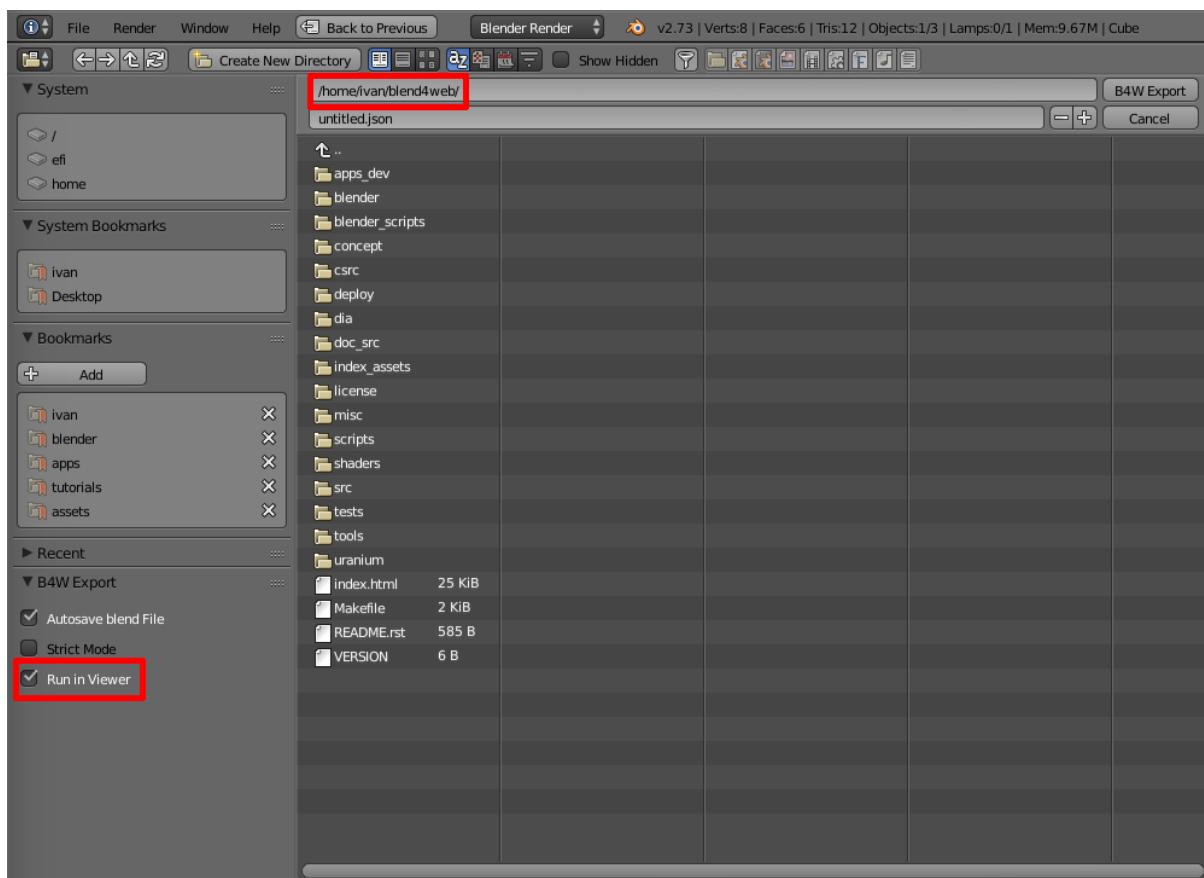
Export Converted Media This option is available for HTML export. When this option is enabled, the converted media files of different formats are written in the HTML file. Using different media files is essential to create cross-browser and cross-platform applications while using HTML export. These files can be created by the [converter](#).



Run in Viewer Automatically launch the Scene Viewer and add the exported scene to it.

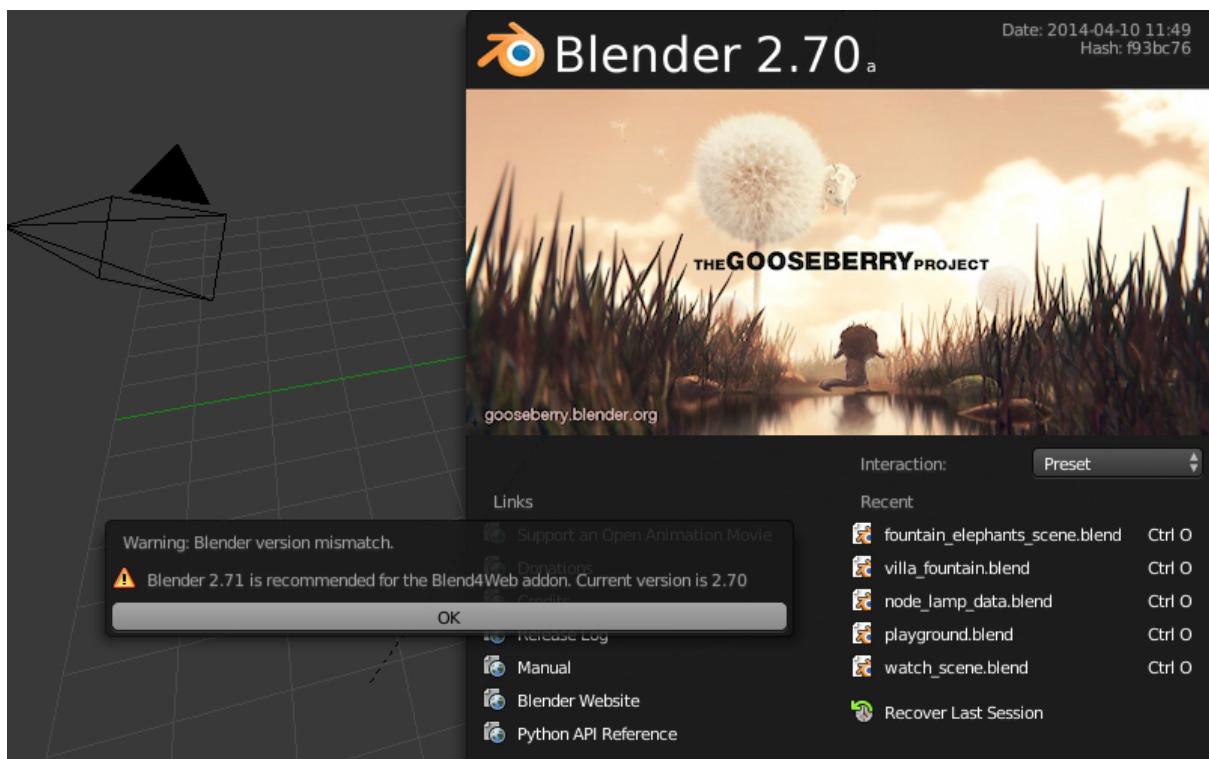
When using the [local development server](#), there is a possibility to open the exported .json scene in the Scene Viewer. To do this, select any path inside the Blend4Web SDK file structure upon export.

A directory inside the SDK should be used for export. If not, this option will not be displayed in the menu. Also, it will not be displayed if the local development server is down.



Initialization Errors

Initialization errors can arise upon installation of the add-on or when a scene is opened in Blender. In this case a dialog window with the error description is showed.



Error message	Cause
Blend4Web initialization error! Addon is not compatible with the PLATFORM platform.	The Blend4Web add-on is not compatible with the PLATFORM platform.
Warning: Blender version mismatch. Blender VER_REQUIRED is recommended for the Blend4Web addon. Current version is VER_CURRENT.	Warning about possible incompatibility with the current Blender version. It is recommended to use VER_REQUIRED Blender version. The current version is VER_CURRENT.

Compatibility Errors

Compatibility errors may arise when trying to view a scene in a browser, in the following cases: if version of the add-on used to export the scene differs from version of the Blend4Web engine which tries to load the scene, or if .bin file does not correspond to the .json file.

Engine version is too old as compared to version of the add-on with which the scene was exported. The scene will not be loaded. We recommend you to use the latest versions of the engine and the add-on.

Error message	Cause
JSON version is too old relative to B4W engine: VER_OLD, required: VER_NEW. Reexport scene with the latest B4W addon to fix it.	Version of the add-on, with which the scene was exported, is too old: VER_OLD. The engine requires: VER_NEW. The scene will not be loaded. We recommend you to reexport the scene using the latest version of the add-on. We also recommend to use the latest version of the engine.
JSON version is a bit old relative to B4W engine: VER_OLD, required: VER_NEW. Some compatibility issues can occur. Reexport scene with the latest B4W addon to fix it.	Version of the add-on, with which the scene was exported, is a bit old: VER_OLD. The engine requires: VER_NEW. The scene will be loaded as usual, however some errors may occur. We recommend you to reexport the scene using the latest version of the add-on. We also recommend to use the latest version of the engine.
B4W engine version is too old relative to JSON. Can't load the scene. Update your engine version to fix it.	Engine version is too old as compared to version of the add-on with which the scene was exported. The scene will not be loaded. We recommend you to use the latest versions of the engine and the add-on.
B4W engine version is a bit old relative to JSON. Some compatibility issues can occur. Update your engine version to fix it.	Engine version is a bit old as compared to version of the add-on with which the scene was exported. The scene will be loaded as usual, however some errors may occur. We recommend you to use the latest versions of the engine and the add-on.
BIN version does not match to JSON version: VER_BIN, required: VER_JSON. Couldn't load the scene. Reexport scene to fix it.	Version of the .bin file is too old relative to .json file: VER_BIN, .json file version is VER_JSON. The scene will not be loaded. We recommend you to reexport your scene.
BIN version does not match to JSON version: VER_BIN, required: VER_JSON. Some compatibility issues can occur. Reexport scene to fix it.	Version of the .bin file is a bit old relative to .json file: VER_BIN, .json file version is VER_JSON. Some incompatibility errors can arise. We recommend you to reexport your scene.

Critical Export Errors

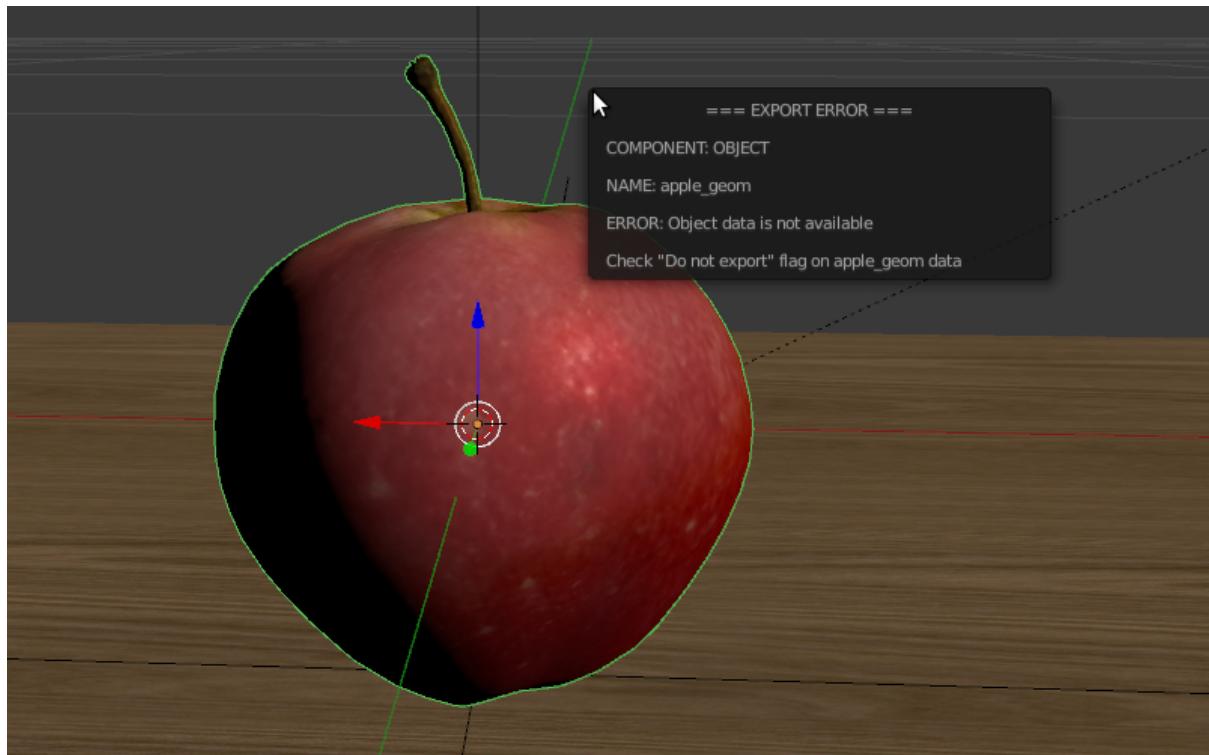
In case of export errors a BLEND4WEB EXPORT ERROR dialog box describing of the problem appears:

COMPONENT - type of component (object, mesh, material, texture etc) that

has caused the export error.

NAME - component name.

ERROR - short description of the occurred problem.



Error message	Cause
Dupli group error; Objects from the GROUP_NAME dupli group on the OBJECT_NAME object cannot be exported	None of the objects in the GROUP_NAME group which were selected for duplication on the OBJECT_NAME object can be exported. Permission to export at least one object of the group, or to remove the duplication of the group is required.
Export to different disk is forbidden	Export to a directory located on a different disk is forbidden
Incompatible objects with a shared mesh; The OBJECT_NAME object has both vertex groups and a shared mesh	Incompatible objects with a shared mesh. Export of an object with both a shared mesh and vertex groups is not allowed. Exceptions: export is possible if an object has the Apply modifiers, Export vertex animation, Export edited normals, Apply Scale and Modifiers options turned on (because in these cases a full copying of meshes occurs).
Incomplete mesh; Material slot is empty	Material slot is empty.
Incomplete vehicle. The NAME vehicle doesn't have any chassis or hull	The modelled NAME vehicle is not complete as it should contain a Chassis or a Hull element.
Incomplete vehicle. The NAME vehicle requires at least one bob	The modelled NAME vehicle is not complete as it should contain at least one Bob element.
Incomplete vehicle. The NAME vehicle requires at least one wheel	The modelled NAME vehicle is not complete as it should contain at least one Wheel element.
Incorrect mesh; Corrupted file: Wrong group indices	The mesh has vertices assigned to the non-existing vertex group.
Incorrect mesh; Corrupted file: Wrong vertice positions	Corrupted file: incorrect vertex coordinate value.
Incorrect mesh; Corrupted file: Wrong normals	Corrupted file: incorrect normal value.
Incorrect mesh; Corrupted file: Wrong tangents	Corrupted file: incorrect tangent value.
Incorrect mesh; Corrupted file: Wrong texture coordinates	Corrupted file: incorrect texture coordinate value.
Incorrect mesh; Corrupted file: Wrong vertex group weights	Corrupted file: incorrect vertex group weight value.

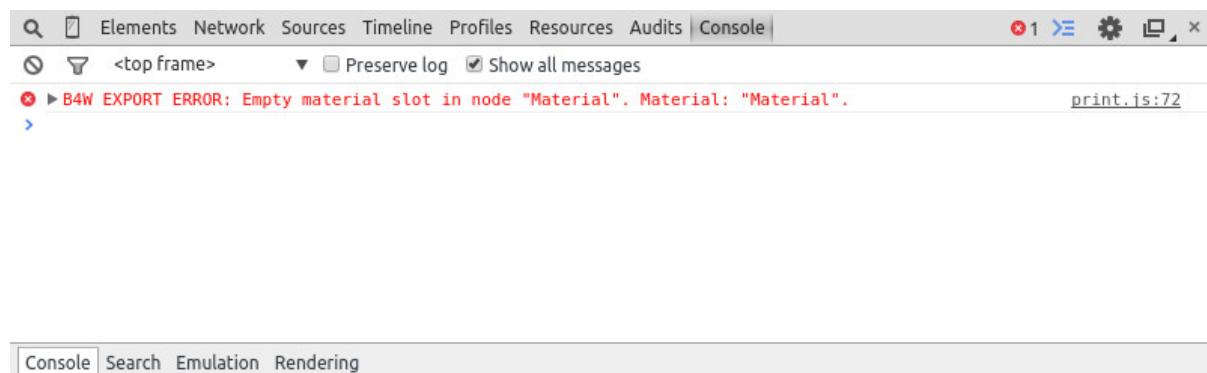
Incorrect mesh; Corrupted file; Wrong vertex color values	Corrupted file: incorrect vertex color value.
Incorrect vertex animation; Object has no vertex animation	The object's vertex animation export option is on, but there is no vertex animation.
Incorrect vertex animation; Unbaked “ANIM_NAME” vertex animation	Vertex animation export is turned on for the mesh, but the ANIM_NAME animation doesn't have any frames.
Loading of resources from different disk is forbidden	Loading of resources from different disk is forbidden.
The material has a normal map but doesn't have any material nodes	The node material uses Normal Mapping, but has no Material node.
The mesh has a UV map but has no exported material	The mesh has a UV map layer but has no material for export.
The mesh has a vertex color layer but has no exported material	The mesh has a vertex color layer but has no material for export.
No such file or directory	The file or directory does not exist.
Object constraint has no target	The Target Object property for the object constraint (on the Object Constraints tab) was not set.
Particle system error; Dupli group isn't specified	Particle system error: no group is selected as a particle..
Particle system error; Dupli object isn't specified	Particle system error: no object is selected as a particle.

Particle system error; Dupli object OBJECT_NAME doesn't export	The OBJECT_NAME object which is selected as a particle can not be exported (the Do not export checkbox is set).
Particle system error; The GROUP_NAME dupli group contains no valid object for export	The GROUP_NAME dupli group which is selected as a particle contains no valid object for export. Either such objects have the Do not export checkbox enabled or the types of the objects are unsuitable. Supported object types: MESH.
Particle system error. Unsupported render type TYPE for the EMITTER/HAIR particles PSYS_NAME on object NAME. Particle system removed.	Particle system error. Unsupported render type TYPE for the EMITTER/HAIR particles PSYS_NAME on object NAME. The particle system has removed.
Particle system error; Wrong dupli object type TYPE_NAME	An object of unsuitable type is selected for the particle. Supported types: MESH.
Permission denied	No access rights to the current directory.
Wrong edited normals count; It doesn't match with the mesh vertices count	The number of edited normals does not match the number of the mesh vertices. Execute Clean Up or Save in the B4W Vertex Normals Editor panel.
Wrong overridden bounding box; Check the mesh's bounding box values	Wrong dimensions are specified when overriding the mesh's BoundingBox: minimum value is greater than maximum value for at least one of the dimensions.
Wrong vertex animation vertices count; It doesn't match with the mesh vertices count for "ANIM_NAME"	Vertex animation export is enabled but the number of vertices in the baked ANIM_NAME animation frames does not match the mesh vertices number. Possible solution is to "re-bake" the animation.

Non-Critical Export Errors

In contrast to the above-listed critical export errors, these errors do not prohibit the export, but can make scenes displayed incorrectly. These messages can be viewed in the browser console (opens with F12) when a scene is loaded. The message looks like this:

B4W EXPORT ERROR: Error message



Error message	Cause
Canvas texture ID NAME already exists. Texture NAME.	This Canvas ID already exists.
Empty canvas texture ID for texture NAME.	Canvas ID is empty.
Empty material slot in node "NAME". Material: "NAME".	Empty material slot in "NAME" node.
Environment map in the "NAME" world texture slot cannot be a movie.	Environment map can not be presented with a video file.
Ignoring LODs after empty LOD for the NAME object.	All LOD objects that follow the empty slot were ignored (in the LOD objects list for the NAME object).
Incomplete mesh NAME; Dynamic grass vertex colors required by material settings	The Dynamic grass size and/or Dynamic grass color options are used by the special terrain material but the mesh has no vertex colors with such names.
Incomplete mesh; Material settings require vertex colors	The Vertex Color Paint option is enabled for the mesh material, but the mesh has no vertex color layers.
Incorrect NLA script, falling back to simple sequential NLA.	Incorrect NLA script, falling back to simple sequential NLA.
Invalid link found in node material. Material: "NAME".	The "NAME" node material contains an incorrect link between nodes.
No image in the NAME texture. [Material: NAME.]	The texture has no image.
No texture for the NAME particle settings texture slot.	No texture in the particle settings' texture slot.
No texture in the NAME world texture slot.	No texture in the NAME world's texture slot.
No texture in the texture slot. Material: NAME.	There is no texture in the material texture slot.
Node material invalid: "NAME". Check sockets compatibility: "FROM_NODE" with "TO_NODE".	Node material error: the input and output types of the link between the FROM_NODE and TO_NODE nodes should match.
Object "NAME" hasn't renderable data. Converted to EMPTY.	An object named "NAME" is degenerate, e.g. has no polygons. The object's type has been changed to EMPTY.

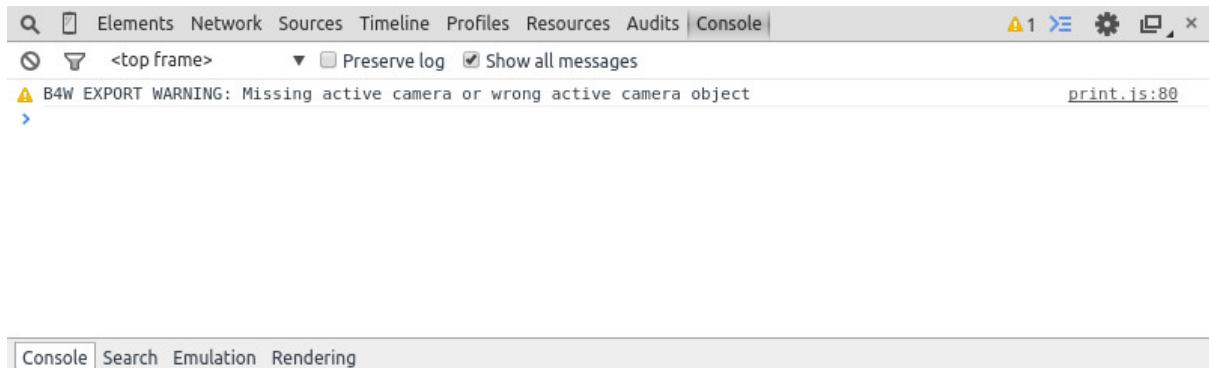
Object “NAME” has the mesh with shape keys. The property “Relative” of mesh has been enabled.	An object named “NAME” has a mesh with shape keys. This mesh has the “Relative” property enabled which is forbidden.
Only 2 UV textures are allowed for a mesh; The mesh has N UVs.	The engine supports up to 2 UV texture layers for each mesh. The number of UV layers for this mesh is N.
Packed media “FILE_NAME” has not been exported to “CONVERTED_FILE_PATH”	The packed media file “FILE_NAME” cannot be converted to “CONVERTED_FILE_PATH”. Please unpack this file and convert it.
Particle system error for “NAME”; The “NAME” vertex color specified in the from field is missing in the last of the “OBJECT_NAME” object’s vertex colors	The NAME vertex color is specified in the from field but it’s not presented in the OBJECT_NAME emitter.
Particle system error for “NAME”; The “NAME” vertex color specified in the to field is missing in the list of the “OBJECT_NAME” object’s vertex colors	The NAME vertex color is specified in the to field but it is not present in the OBJECT_NAME object which is selected as a particle.
Particle system error for “NAME”; The “NAME” vertex color specified in the “to field is missing in the “OBJECT_NAME” object (“GROUP_NAME” dupli group)	The NAME vertex color is specified in the to field but it is not present in the OBJECT_NAME object of the GROUP_NAME group which is selected as a particle.
Particle system error for object “NAME”. Invalid dupli object “OBJECT_NAME”.	Particle system error for the object NAME. Invalid dupli-object OBJECT_NAME.
Sound file is missing in the SPEAKER object “NAME”. Converted to EMPTY.	The speaker has no sound attached. The object’s type has been changed to EMPTY.
The main scene NAME can not be rendered by another scene. Material NAME has been removed.	The main scene NAME can not be rendered by another scene. The material NAME has been deleted.
The NAME action has decimal frames. Converted to integer.	The NAME action has decimal frames. Converted to integer.
The NAME armature modifier has a proxy object as an armature. Modifier removed.	An armature modifier has a proxy object as an armature.
The NAME armature modifier has no armature object or it is not exported. Modifier removed.	The NAME Armature modifier has no armature object or it is not exported. Modifier removed.

The NAME curve modifier has no curve object. Modifier removed.	The NAME curve modifier has no object. Modifier removed.
The NAME curve modifier has unsupported curve object. Modifier removed.	The NAME curve modifier has unsupported object. Modifier removed.
The NAME object has the NAME armature modifier and a vertex animation. Modifier removed.	The NAME object has both vertex animation and an armature modifier which is not supported. As a result, the modifier has been removed.
The NAME LAMP node has no lamp object. Material: NAME.	Wrong object specified in the NAME LAMP node.
The NAME node is not supported. The NAME material will be rendered without nodes. Material: NAME.	The engine does not support the node with this name, and so the node material will be turned off. Often this happens when Cycles nodes are used.
The NAME object has NAME armature modifier which references the wrong group. Modifier removed.	An object should be in the same group as an armature, or both these objects should be explicitly present in the scene.
TEXTURE_TYPE isn't supported, WORLD_NAME	TEXTURE_TYPE texture type isn't supported for world WORLD_NAME
Using B4W_REFRACTION node NODE_NAME with incorrect type of Alpha Blend. Material: NAME.	A node material with incorrect Alpha Blend property is used. Alpha sort, Alpha blend and Add are allowed when using a "REFRACTION" node.
Wind bending: not all vertex colors exist for "NAME". Properties were set to default values.	Wind bending parameters setup: all specified vertex color layers should exist.
Wind bending: vertex colors weren't properly assigned for "NAME". Properties were set to default values.	Wind bending parameters setup: it's required to specify the names of either all vertex color layers (Main stiffness (A), Leaves stiffness (R), Leaves phase (G), Overall stiffness (B)), or of the main one only (Main stiffness (A)), or of none of them.
Wrong "Height Map" input for the "NAME" B4W_PARALLAX node. Only link from the TEXTURE node with a non-empty texture is allowed.	Wrong data were passed to the "Height Map" input of the NAME B4W_PARALLAX node. Only the output from a non-empty TEXTURE node is allowed.
Wrong texture coordinates type in texture NAME. [Material: NAME.]	The following coordinate types are supported for image textures: UV, Normal and Generated.

Other Messages

These messages can be viewed in the browser console (opens with F12) when a scene is loaded. The message looks like this:

B4W EXPORT WARNING: Export message which requires the user's attention



Error Message	Cause
Missing active camera or wrong active camera object	There is no active camera on the scene (Camera property on the Scene tab).
Missing world or wrong active world object	There should be at least one world datablock in the scene.
NAME particle settings has the NAME texture rendering a scene. It has been replaced by the default texture.	The particle settings datablock NAME contains the texture NAME being used for rendering a scene into. This texture has been replaced by a default texture.
The action NAME has no fcurves.	The action NAME has no fcurves.
The "NAME" camera has unsupported PANORAMIC type. Changed to PERSPECTIVE type."	Panoramic cameras are not supported. Perspective mode is used instead.

Add-on Translations

There is the possibility to translate the add-on to a language supported by Blender. In order to do this, rename the file “empty.po”, which located in the directory `SDK/blender_scripts/addons/blend4web/locales`, to one of the names in the following table:

File name	Language
ru_RU.po	Russian
ja_JP.po	Japanese
en_US.po	English
Continued on next page	

Table 8.1 – continued from previous page

File name	Language
nl_NL.po	Dutch
it_IT.po	Italian
de_DE.po	German
fi_FI.po	Finnish
sv_SE.po	Swedish
fr_FR.po	French
es.po	Spanish
ca_AD.po	Catalan
cs_CZ.po	Czech
pt_PT.po	Portuguese
zh_CN.po	Simplified Chinese
zh_TW.po	Traditional Chinese
hr_HR.po	Croatian
sr_RS.po	Serbian
uk_UA.po	Ukrainian
pl_PL.po	Polish
ro_RO.po	Romanian
ar_EG.po	Arabic
bg_BG.po	Bulgarian
el_GR.po	Greek
ko_KR.po	Korean
ne_NP.po	Nepali
fa_IR.po	Persian
id_ID.po	Indonesian
sr_RS@latin.po	Serbian Latin
ky_KG.po	Kyrgyz
tr_TR.po	Turkish
hu_HU.po	Hungarian
pt_BR.po	Brazilian Portuguese
he_IL.po	Hebrew
et_EE.po	Estonian
eo.po	Esperanto
es_ES.po	Spanish from Spain
am_ET.po	Amharic
uz_UZ.po	Uzbek
uz_UZ@cyrillic.po	Uzbek Cyrillic
hi_IN.po	Hindi
vi_VN.po	Vietnamese

Then open this file and edit/translate it.

When translations are ready, you may contact us to include them as part of the add-on.

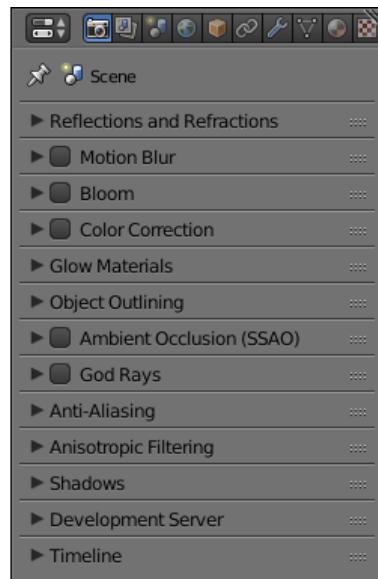
Scene Settings

Table of Content

- Scene Settings
 - Render Panel
 - * Reflections and Refractions
 - * Motion Blur
 - * Bloom
 - * Color Correction
 - * Glow Materials
 - * Object Outlining
 - * Ambient Occlusion (SSAO)
 - * God Rays
 - * Anti-Aliasing
 - * Shadows
 - * Development Server
 - * Timeline
 - Scene Panel
 - * Scene
 - * Units
 - * Audio
 - * Logic Editor
 - * NLA
 - * Meta Tags
 - * Physics
 - * Batching
 - * Objects Selection
 - * Anchors
 - * Export Options
 - World Panel
 - * Preview
 - * World
 - * Environment Lighting
 - * Mist
 - * Procedural Sky
 - * Export Options

All the parameters that define the look and behavior of the whole scene (and not just a single object) are found on the three panels: the Render panel, the Scene panel and the World panel.

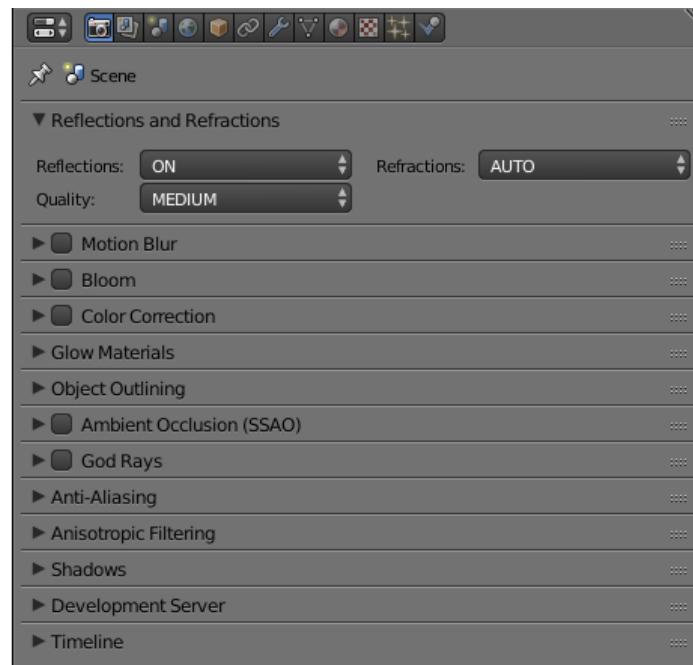
Render Panel



All scene parameters that concern the image rendering are found on this panel.

Reflections and Refractions

Reflection and refraction effect settings.



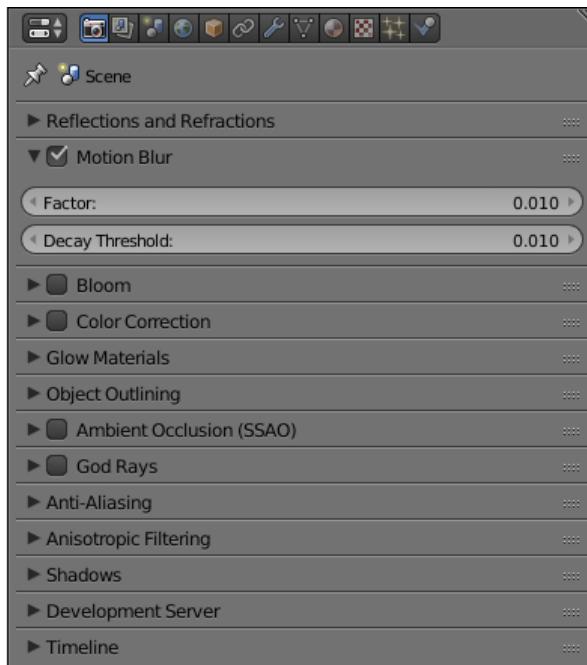
Reflection Effect settings. Can be set to ON, OFF or AUTO. Set to ON by default.

Refraction Effect settings. Can be set to ON, OFF or AUTO. Set to AUTO by default.

Quality Quality settings for the reflection effect. Can be set to HIGH (the highest reflection quality), MEDIUM (medium quality) or to LOW (the lowest quality). Set to MEDIUM by default.

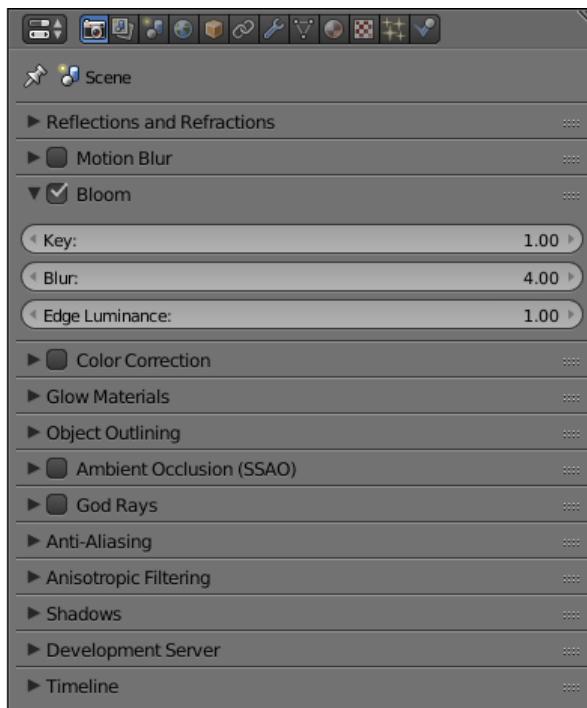
Motion Blur

Motion blur settings. Described thoroughly in its own section.



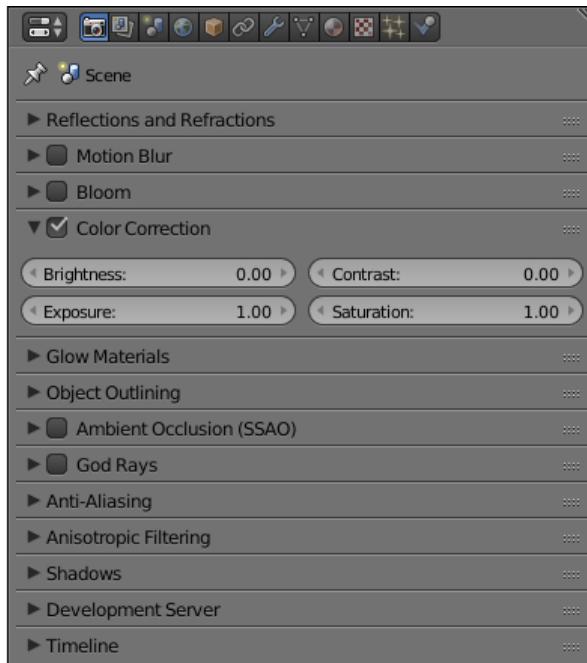
Bloom

Bloom effect settings. Described thoroughly in its own section.



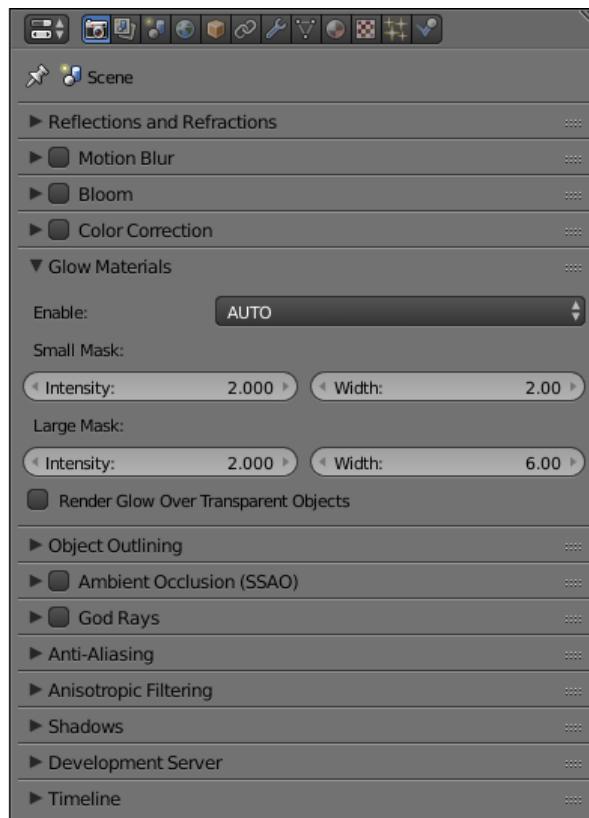
Color Correction

Color correction settings. Described thoroughly in its own section.



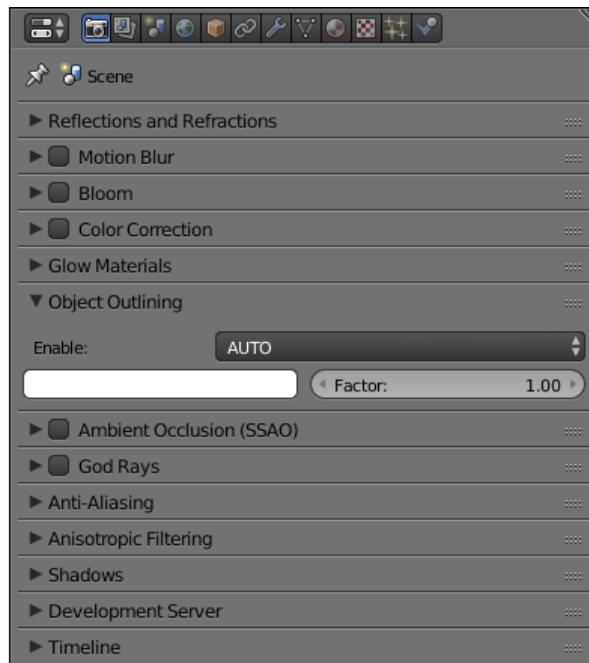
Glow Materials

Glow Material effect settings. Described thoroughly in its own section.



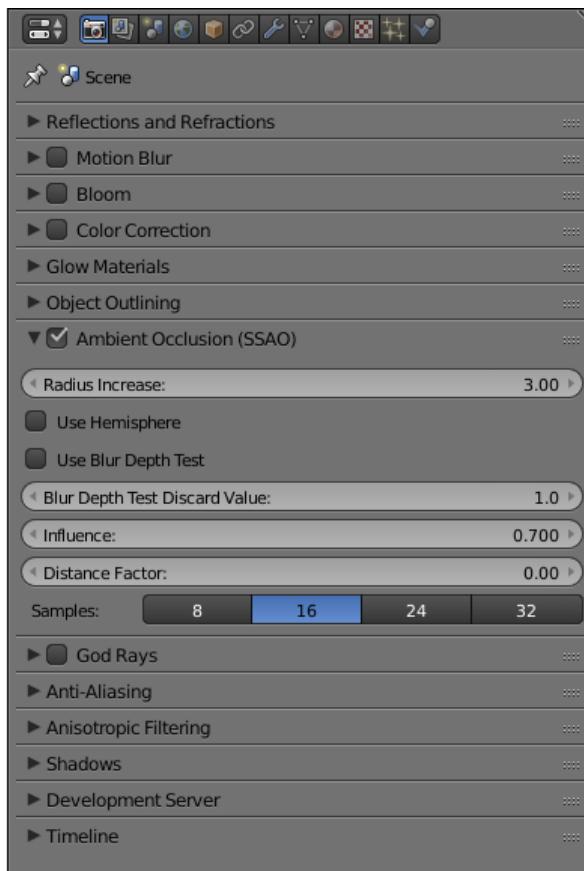
Object Outlining

Outlining effect settings. Described thoroughly in [its own section](#).



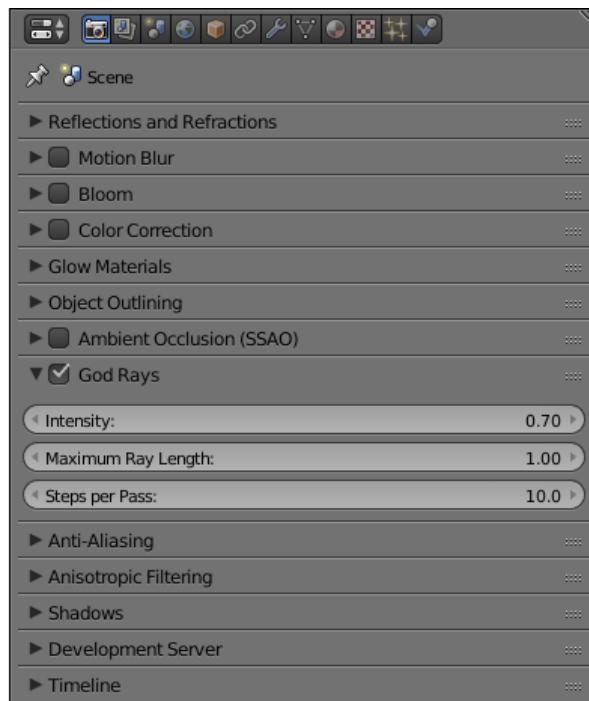
Ambient Occlusion (SSAO)

Screen-space ambient occlusion (SSAO) settings. Described thoroughly in its own section.



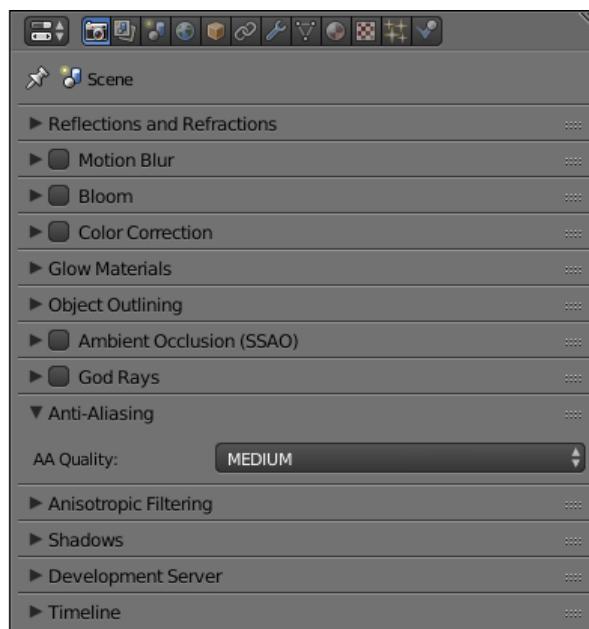
God Rays

God Rays effect settings. Described thoroughly in its own section.



Anti-Aliasing

Anti-Aliasing settings. Described thoroughly in [its own section](#).



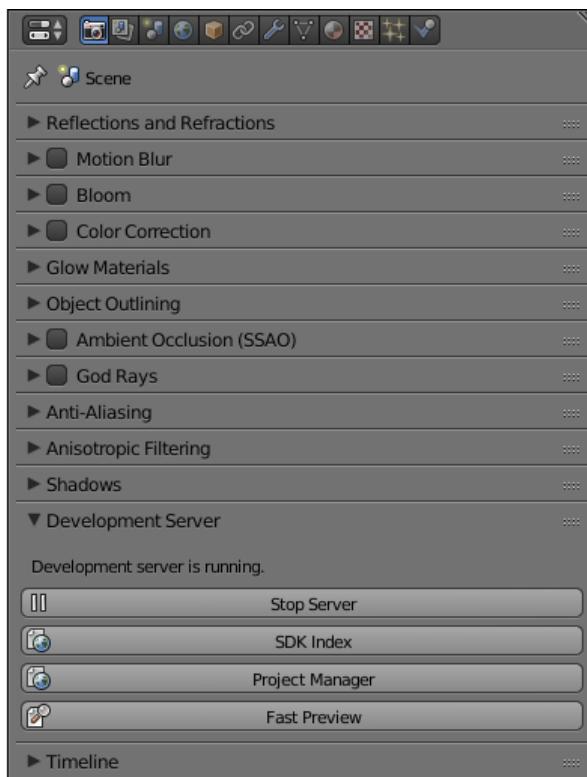
Shadows

Shadows settings. Described thoroughly in [its own section](#).



Development Server

Development Server settings. Described thoroughly in [its own section](#).



Timeline

Timeline settings.

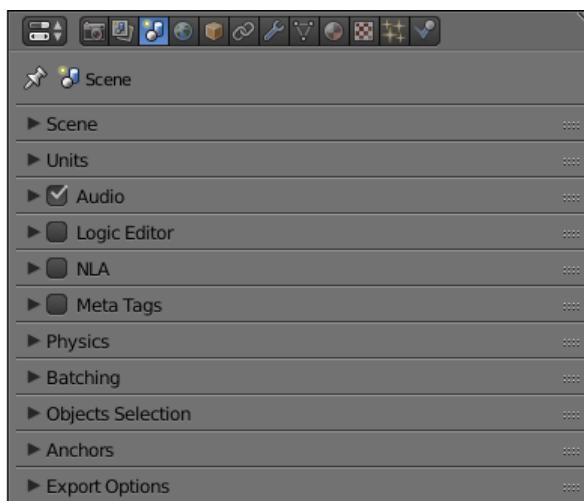


Start Frame The first frame of the timeline. Set to 1 by default.

End Frame The last frame of the timeline. Set to 250 by default.

Frame Rate Number of the frames per second. Set to 24 by default. This parameter only affect the animation playback speed (not the scene itself).

Scene Panel



The settings that concern scene behavior, including audio, physics and animation settings, are found on this panel.

Scene

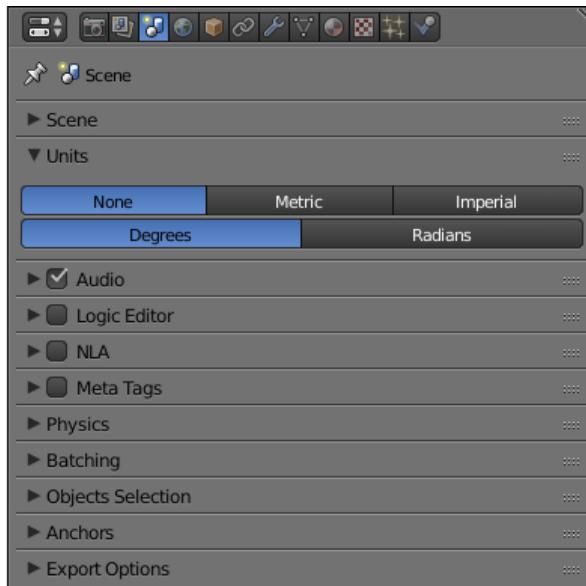
Scene settings.



Camera A camera that is used to render the scene.

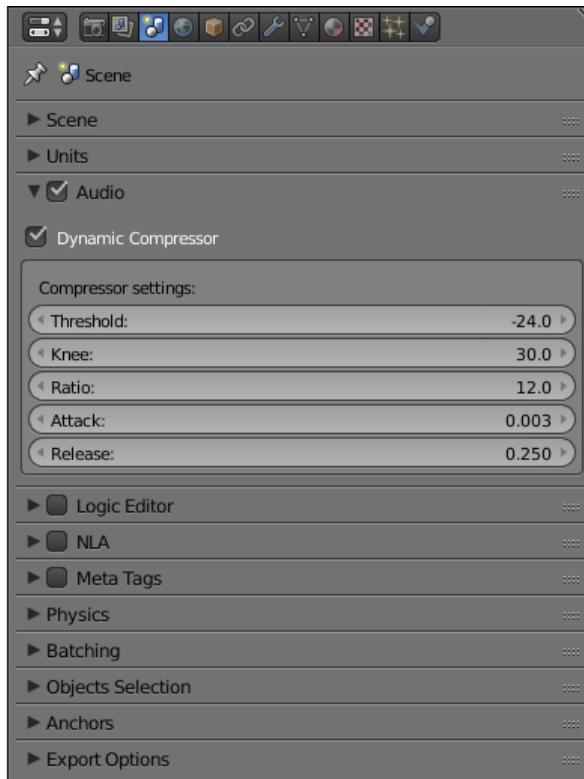
Units

Sets the measurement units used in the scene.



Audio

Audio settings. Enabled by default.



Dynamic Compressor Compress audio signal's dynamic range. This feature can be used to make the sound more rich and even. Disabled by default.

Threshold If the amplitude of the audio signal exceeds the value specified by this parameter, the compressor will reduce its level. Set to -24 dB by default.

Knee The interval below the threshold where the response curve switches to the decreasing mode. Set to 30 by default.

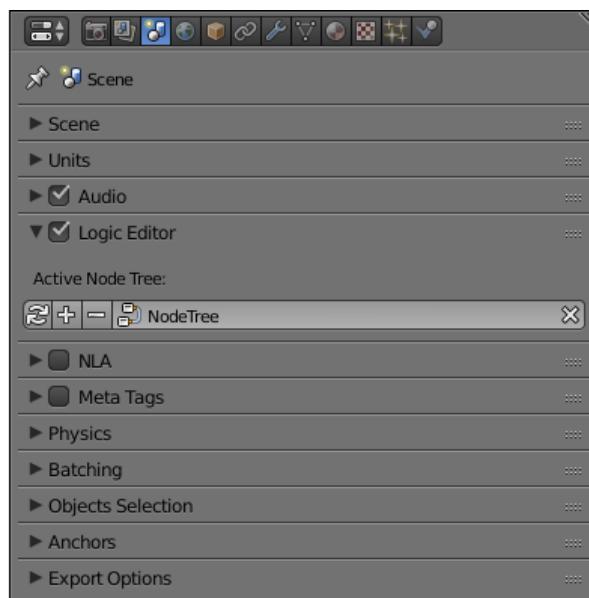
Ratio Amount of gain reduction. Set to 12 by default.

Attack Time (in seconds) that takes the compressor to reduce gain by 10 dB. Set to 0.003 by default.

Release Time (in seconds) that takes the compressor to increase gain by 10 dB. Set to 0.25 by default.

Logic Editor

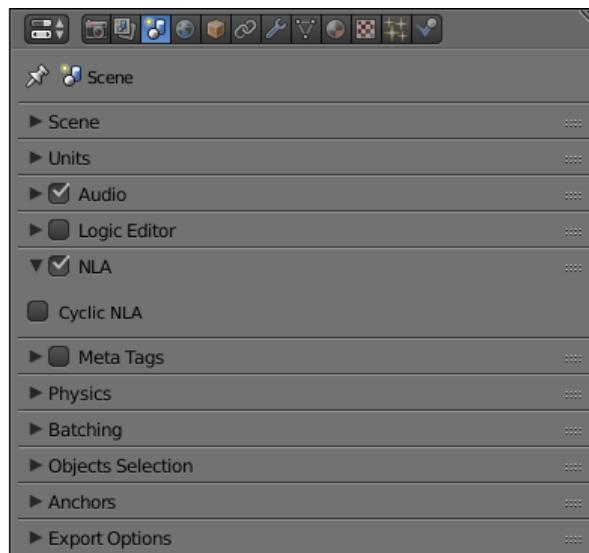
Settings for the use of the logic node trees (created in the logic editor) in the scene. Disabled by default.



Active Node Tree Node tree that is used in the scene's playback.

NLA

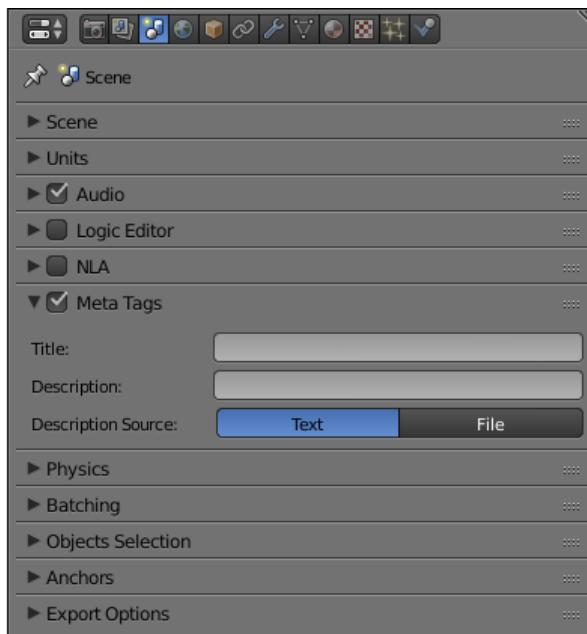
Non-Linear Animation playback settings. Disabled by default.



Cyclic NLA If this parameter is enabled, NLA animation will be repeated after it is finished.

Meta Tags

Application's meta tags.



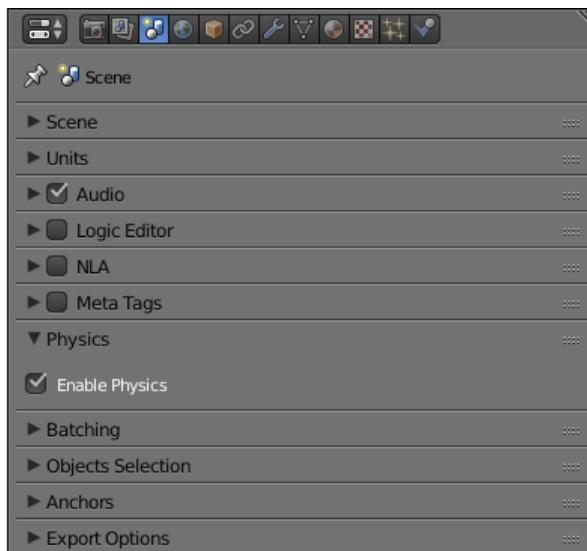
Title The title of the application.

Description The description of the application. Can be a simple text or a link to a text file (if the Description Source parameter is set to the File value).

Description Source The source of the application's description. The description can be loaded from a file or specified directly in the Description field. This parameter can have one of the two values, Text and File, and it is set to Text by default.

Physics

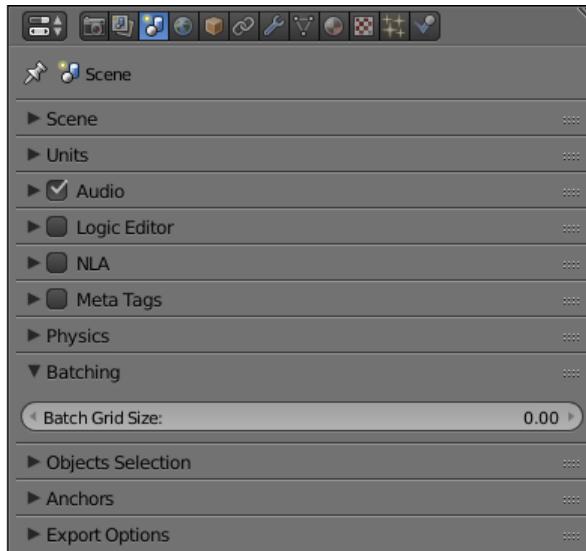
Physics settings.



Enable Physics Allow using physics in the application. Enabled by default.

Batching

The size of the grid used to batch the object with the same materials can be specified here. This parameter can be used for optimization purposes.

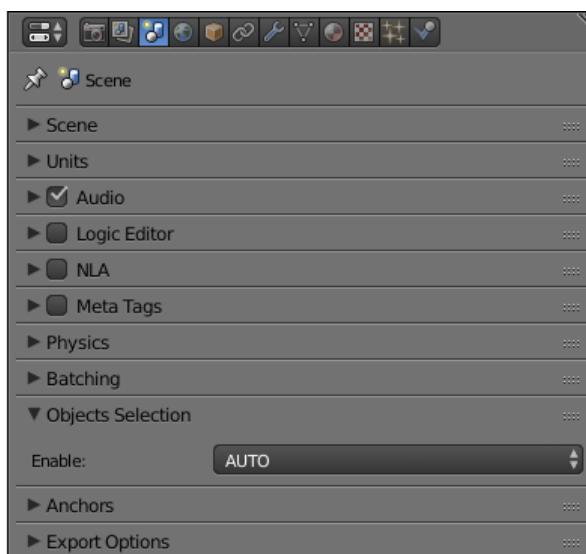


Batch Grid Size The size of the grid in meters. Set to zero by default (in such case, the grid isn't used).

Objects Selection

Object selection settings. Objects can be selected both with the API function `pick_object()` and with the logic nodes.

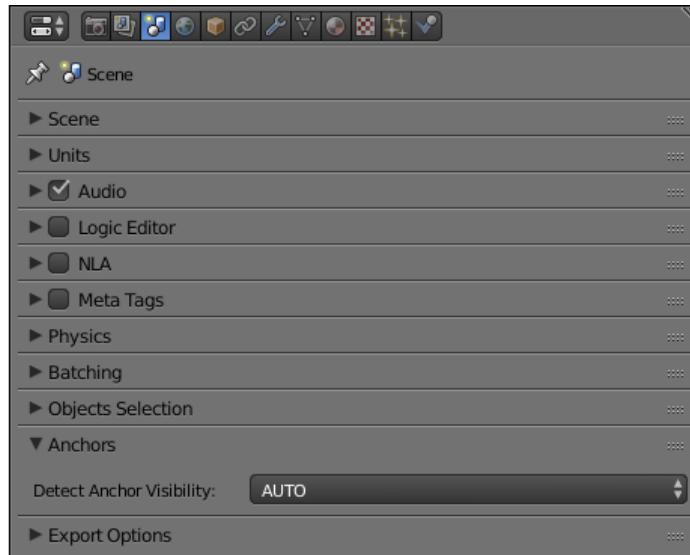
Note: In the `scene` viewer, selection is enabled by default. You can turn it off in the Tools & Debug panel.



Enable The parameter that defines if the object can or can't be selected. It can have ON, OFF or AUTO value. Set to AUTO by default.

Anchors

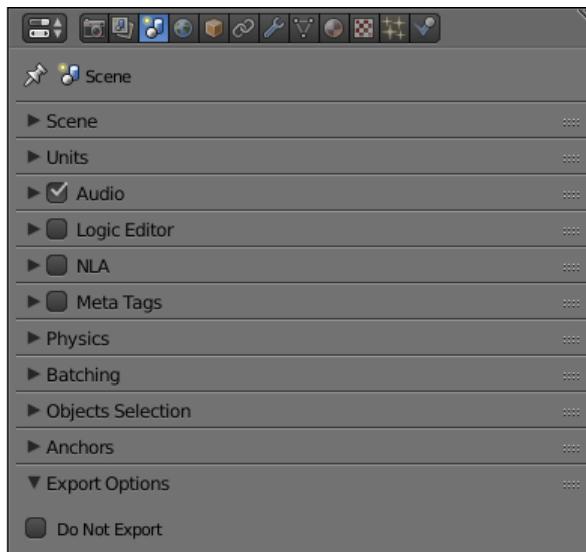
Anchor visibility detection settings.



Detect Anchors Visibility Anchor visibility detection parameter. Can have ON, OFF or AUTO value. Set to AUTO by default.

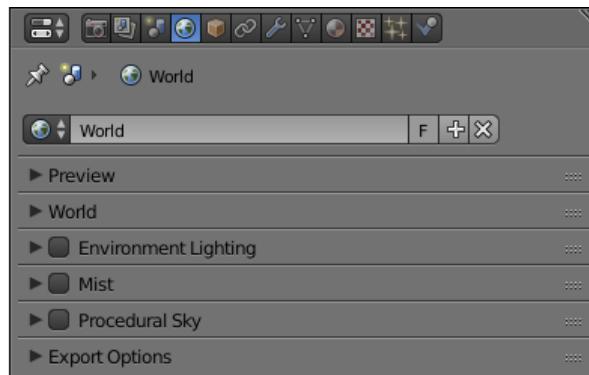
Export Options

Scene settings export parameters.



Do Not Export If this parameter is enabled, scene settings will be ignored during export.

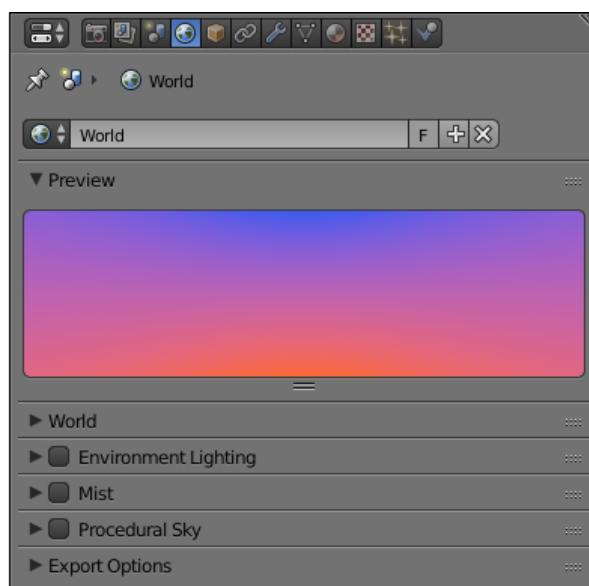
World Panel



Environment settings such as settings for sky, mist and such are found on this panel.

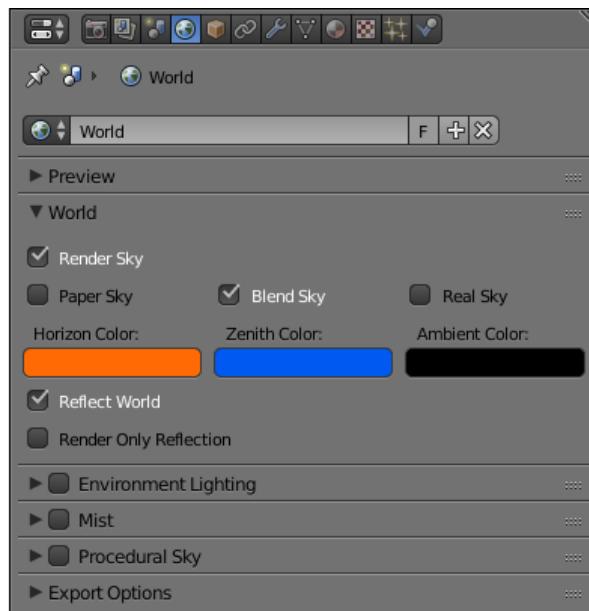
Preview

Environment preview.



World

Sky settings.



Render Sky If this parameter is enabled, the engine will render sky in the scene.

Paper Sky If this parameter is enabled, sky gradient will always be drawn from the top of the screen to the bottom, regardless of the camera's position and angles.

Blend Sky Smooth transition between the horizon and zenith colors.

Real Sky Sky rendering with the horizon affected by the camera angle.

Horizon Color Sky color at the horizon.

Zenith Color Sky color in the zenith.

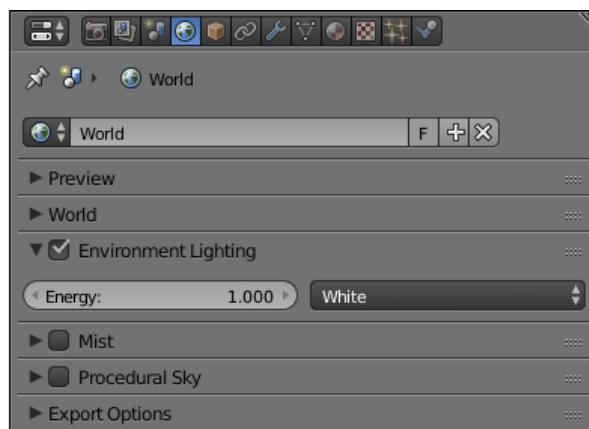
Ambient Color The color of the ambient lighting.

Reflect World Render the sky while rendering reflections.

Render Only Reflection Render the sky only while rendering reflections.

Environment Lighting

Environment lighting settings. Described thoroughly in [their own section](#).



Mist

Mist settings.



Minimum Minimum intensity of the mist. Set to zero by default.

Depth At this distance from the camera the mist effect reaches maximum intensity. Set to 25 by default.

Start The mist effect starts to appear at this distance from the camera. Set to 5 by default.

Height This parameter specifies how fast mist intensity decreases as the camera's altitude increases. Set to 0 by default.

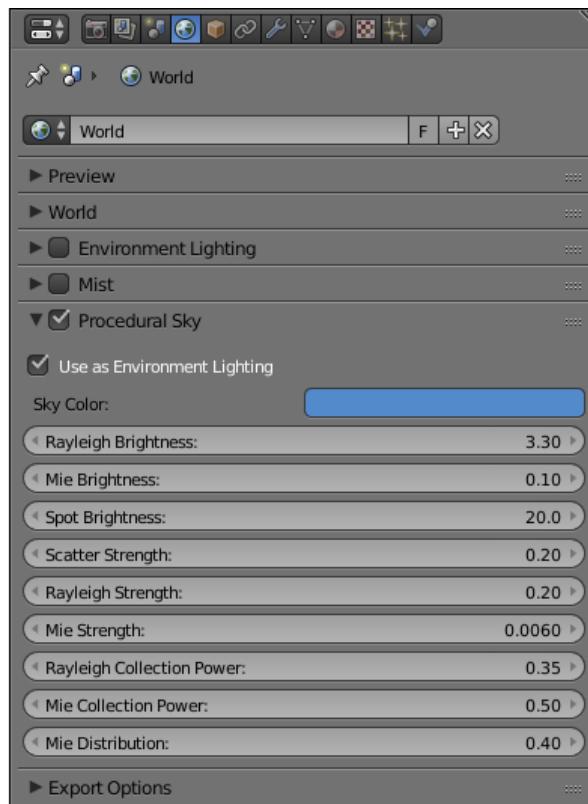
Fall Out This parameter specifies the rule, according to which the density of the mist changes between the borders (specified by the Start and Depth parameters). Can have one of the following values: Quadratic, Linear, Inverse Quadratic. Set to Quadratic by default.

Use Custom Colors Can be used to set the color of the mist. Enabled by default. If this parameter is disabled, standard (0.5, 0.5, 0.5) color will be used.

Fog Color The color of the mist. Can be changed, if the Use custom colors parameter is enabled. Light gray (0.5, 0.5, 0.5) color is used by default.

Procedural Sky

Procedural sky settings. Described thoroughly in [their own section](#).



Export Options

Environment parameters export settings.



Do Not Export If this parameter is enabled, environment settings will be ignored during the export.

Objects

Table of Contents

- Objects
 - Types
 - Static and Dynamic Objects
 - Settings
 - * Object Tab
 - * Physics Tab
 - Anchor Settings
 - Object Transform API
 - Get Object API
 - Object Selection
 - Copying Objects (Instancing)
 - * Making a Simple Copy
 - * Making a Deep Copy
 - * Removing Objects
 - Quaternions
 - * Quaternion Example
 - Differences Between Coordinate Systems of Blender and Blend4Web
 - Moving via TSR Vectors

Objects are intended to position components of different types (meshes, cameras, lamps etc) in a 3D scene space.

Types

The engine supports objects of the following types:

- meshes (mesh)
- camera
- lamp

- empty
- armature
- speaker
- curve
- text
- metaball
- surface

During the scene export, CURVE, TEXT, METABALL and SURFACE type objects are converted into MESH type objects.

Static and Dynamic Objects

An object can be either static or dynamic.

Static objects are objects that cannot be moved, animated or changed in any other way while running the application. For performance purposes, such objects can be merged together if they have the same material.

Dynamic objects can be moved, animated or changed in other ways while running the application. They can also interact with other objects, including static ones. Dynamic objects are never combined with each other or with static objects.

Only MESH and EMPTY type objects can be either static or dynamic. All other object types, such as CAMERA and ARMATURE, are always dynamic.

Static MESH objects are rendered much faster than the dynamic ones, so, for better performance, it is advised to keep the number of dynamic meshes to a minimum. Objects of any other type, both static and dynamic, do not significantly affect performance.

The objects which have animation, physics or a parent, which is a dynamic object, are considered dynamic, as well as the objects controlled by the following logic nodes:

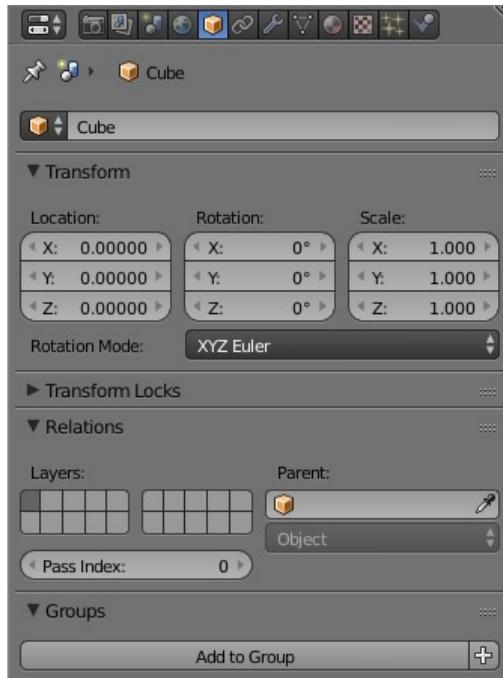
- Play Animation
- Transform Object
- Move To
- Inherit Material

API methods that concern object movement, copying and animation (both object and node material) can only be applied to dynamic objects. In order to make the movement of the object without dynamic settings possible, it is necessary to activate Force Dynamic Object option in its settings.

Settings

The following is supported for all types of objects: transform, data reference, parent object, group membership and a set of the Blend4Web's special properties.

Object Tab



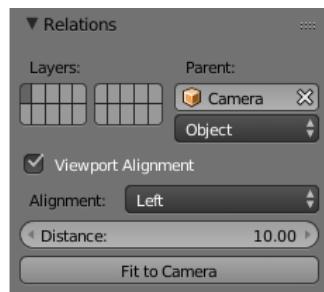
Transform > Location Position coordinates.

Transform > Rotation Rotation angles. For the object rotation all available modes can be used (Rotation Mode). However only Quaternion (WXYZ) and XYZ Euler are supported for [object animation](#).

Transform > Scale Scaling. All 3 components (x, y, z) should be the same. Scaling for physics objects is not supported.

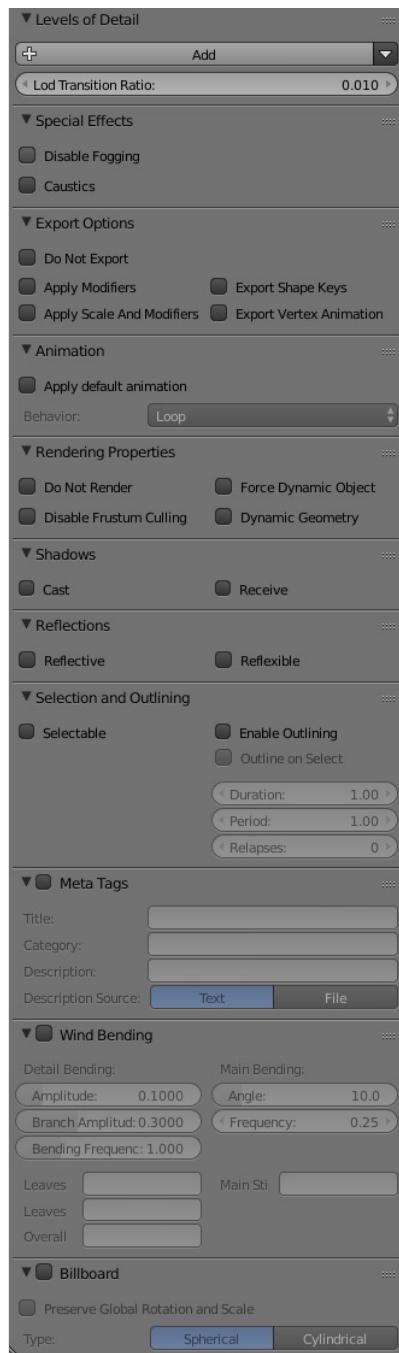
Relations > Parent Reference to the parent object.

If the parent object is a camera, Viewport Alignment settings are available.



These settings can be used to align the object to an active camera. They are described in the [Camera chapter](#).

Groups Objects' groups to which this object belongs.



Levels of Detail > LOD Transition Ratio Parameter for smoothing the switching between the LOD objects. It defines the additional distance at which the LOD objects are still rendered before getting replaced by the next LOD objects. Assigned for the main object. Measured in fractions of the object's bounding sphere radius.

Special Effects > Disable Fogging Disable fog for the object.

Special Effects > Caustics The object will render caustics effects from the adjacent water.

Export Options > Do Not Export Do not export this object.

Export Options > Apply Modifiers Apply the object's modifiers upon export. If the SKIN modifier is used we recommend to apply it before the export because it resets vertex color and UV layers which may result in errors.

Export Options > Apply Scale and Modifiers Upon export, apply scale and modifiers for the object.

Export Options > Export Vertex Animation Export previously created and saved vertex animation. Applicable for MESH type objects only.

Export Options > Export Shape Keys Export shape keys. Applicable to MESH type objects only.

Note: The following properties are mutually exclusive: Apply Modifiers, Apply Scale and Modifiers, Export Vertex Animation and Export Shape Keys.

Animation > Apply Default Animation Upon loading into the engine start playback of the animation assigned to the object.

Animation > Animation Blending Only for armature objects. Allows blending between skeletal animations.

Animation > Behavior Animation behavior when the last frame is reached: Finish Stop - stop, Finish Reset - stop and go to the zero frame, Loop - repeat forever.

Rendering Properties > Do Not Render Disable object rendering (for example useful for a physics object).

Rendering Properties > Disable Frustum Culling Disable frustum culling optimization.

Rendering Properties > Force Dynamic Object Force the object to become a [dynamic object](#).

Rendering Properties > Dynamic Geometry Allow overriding of the object's geometry through Blend4Web API.

Shadows > Cast The object will cast shadows.

Shadows > Cast Only The object will cast shadows but will remain invisible itself. Becomes available after enabling Shadows > Cast.

Shadows > Receive The object will receive shadows from other adjacent objects.

Reflections > Reflexible When enabled the object is reflected in the dynamic mirror surfaces.

Reflections > Reflexible Only The object will be reflected but will remain invisible itself. Becomes available after enabling Reflections > Reflexible.

Reflections > Reflective When enabled the object surface reflects other objects.

Reflections > Reflection Plane Text field for name of an empty object which defines the reflection plane. Becomes available after enabling Reflections > Reflective.

Selection and Outlining > Selectable Enable [object selection](#) with the mouse or another input device.

Selection and Outlining > Enable Outlining Enable [outline glow](#) for the object.

Meta Tags Interface for adding meta tags to the object:

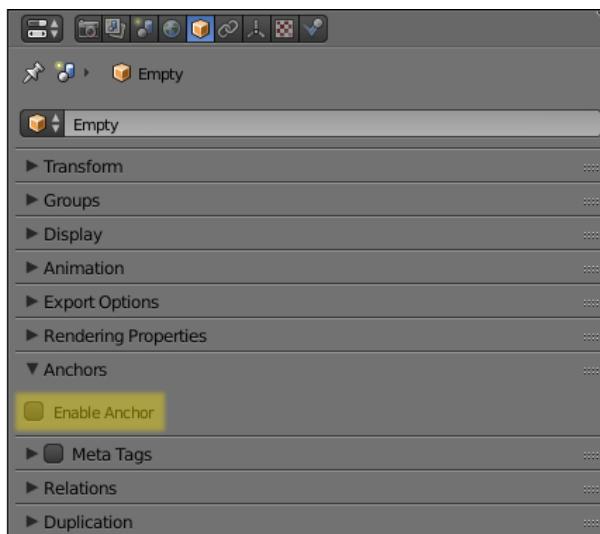
Meta Tags > Title Object's title.

Meta Tags > Category Object's category.

Meta Tags > Description Description for the object. Depending on Description Source, this field accepts either description text itself, or the name of a file where this description is contained.

Meta Tags > Description Source Source type for the description: text or text file.

Anchors > Enable Anchor This parameter enables interface for adding anchors (2D tags) to objects. Available for EMPTY objects only. Described in the [corresponding section](#).



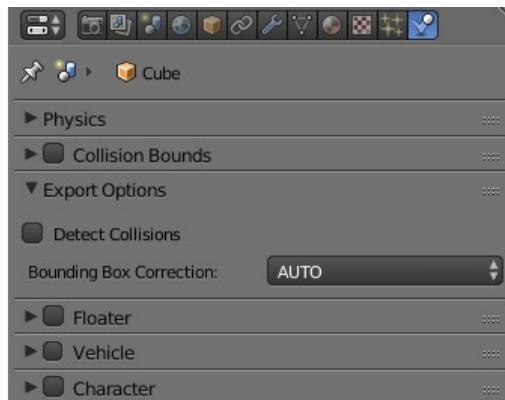
Wind Bending Enables wind bending procedural animation. Thoroughly described at the [outdoor rendering](#) page.

Billboard Use the object as a billboard (i.e. automatically orient relative to the camera).

Billboard > Preserve Global Orientation and Scale Take into account rotation and scale of the billboard object (in the world space). The object will be directed toward the camera with its side which is visible when viewing along the Y axis in Blender. Becomes available after enabling the Billboard checkbox.

Billboard > Billboard Type Billboard orientation mode. Spherical (by default) - the object is always oriented with one side toward the camera, regardless of view angle, Cylindrical - similar to Spherical, but rotation is limited to Blender's world Z axis. Becomes available after enabling Billboard

Physics Tab



Detect Collisions Activate the object's physics.

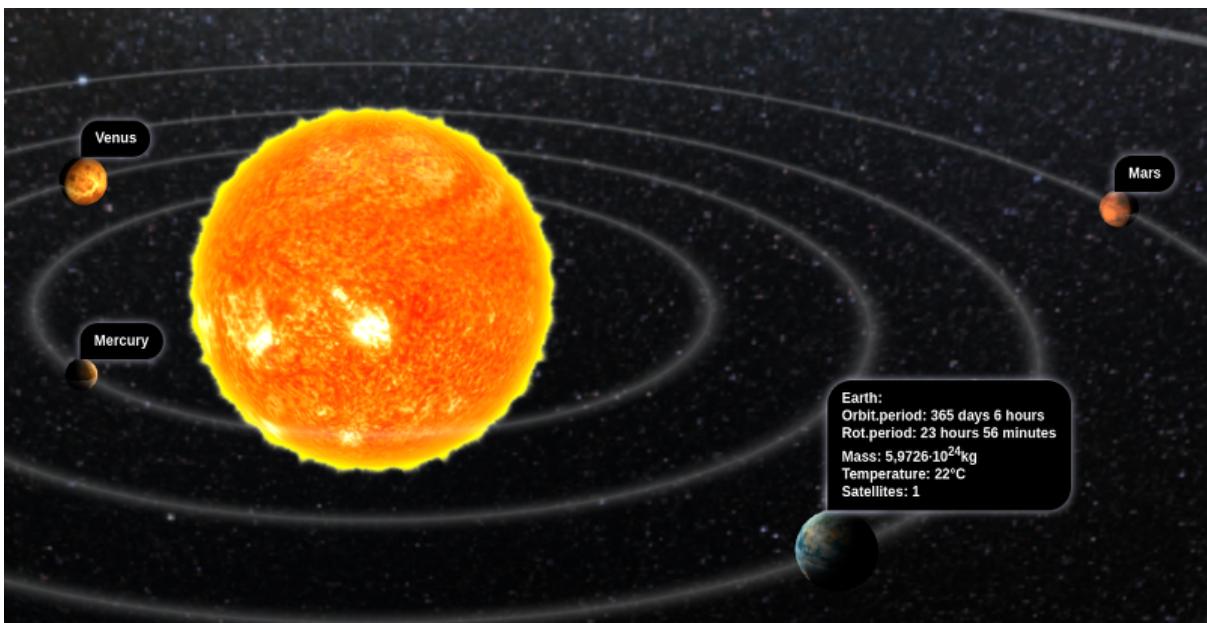
Floating Make the object floating. The settings for floating objects are described in detail in the [physics](#) section.

Vehicle Use the object as part of a vehicle. The vehicle settings are described in detail in the [physics](#) section.

Character Use the object for character physics. The character settings are described in detail in the [physics](#) section.

Anchor Settings

Anchors can be used to attach annotations to 3D objects. The annotation is displayed near the object regardless of the camera position and even follows it throughout the animation.



Annotations can be created entirely in Blender. All you need to do is to place an Empty object in the required position and enable the Anchor property. The text for the annotations can be assigned in the Title and Description fields on the Meta Tags panel.



Enable Anchor This parameter enables the interface for adding anchors (2D tags) to objects. This is available for EMPTY objects only.

Type Anchor type

- Annotation - content is obtained from the [meta tags](#) assigned to the object and displayed in the standard section.
- Custom Element - an arbitrary HTML element from the current web page is used as an anchor.
- Generic - an anchor's position can be detected using the anchors API module.

Default value is Annotation.

HTML Element ID This specifies the ID of the HTML element that will be used as the anchor. This is available only if the Type parameter is set to Custom Element.

Detect Visibility Detect whether the anchor object is overlapped by other objects. This is disabled by default. Turning this option on decreases performance and should be used only when necessary.

Max Width This parameter limits the expanding info window by a predefined value (measured in CSS pixels). This is available only if the Type parameter is set to Annotation, and it is set to 250 by default.

Object Transform API

Note: Make sure that the object you are trying to transform is a [dynamic object](#).

Use the following methods of the `transform` module to move objects in the engine:

`get_translation()` Get the coordinates of the object's center in world space. The method with a single argument returns a new vector (i.e. this is a non-optimized option) while the method with two arguments requires an additional vector to write the result down.

`get_translation_rel()` Similar to the `get_translation()` method, but if this object has a parent, the obtained coordinates are measured in the parent's space.

`set_translation()`, `set_translation_v()` Move the object's center into the specified location. The first method takes separate coordinates as arguments while the second one takes a three-component vector (Array or Float32Array).

`set_translation_rel()`, `set_translation_rel_v()` Similar to `set_translation()` and `set_translation_v()`, but if this object has a parent, the set coordinates are measured in the parent's space.

`get_rotation()` Get the object's rotation quaternion in world space. Similar to `get_translation()`, there are two options for calling this function.

`get_rotation_rel()` Get the object's rotation quaternion measured in its parent's space. Similar to `get_translation_rel()`, there are two options for calling this function.

`set_rotation()`, `set_rotation_v()` Set the object's rotation quaternion in world space. The first function takes separate coordinates as arguments while the second one takes a four-component vector (Array or Float32Array).

`set_rotation_rel()`, `set_rotation_rel_v()` Set the object's rotation quaternion measured in its parent's space. The first function takes separate coordinates as arguments while the second one takes a four-component vector (Array or Float32Array).

`get_scale()` Get the object's scale in world space.

`get_scale_rel()` Get the object's scale in its parent's space.

`set_scale()` Set the object's scale in world space. Unity corresponds to the original scale of the object. Values less than unity mean scaling down, bigger than unity - scaling up. Note that not all objects can be scaled. Particularly, scaling is not allowed for physics objects.

`set_scale_rel()` Set the object's scale in its parent's space.

`set_rotation_euler()`, `set_rotation_euler_v()` Set the object's rotation using Euler angles. An intrinsic YZX rotation system is used (that means the angles follow in the YZX order and the origin of coordinates rotates and takes up a new position for every angle).

`set_rotation_euler_rel()`, `set_rotation_euler_rel_v()` Set the object's rotation using Euler angles measured in its parent's space.

`move_local()` Move the object relative to its original position (in local space).

`rotate_x_local()`, `rotate_y_local()`, `rotate_z_local()` Rotate the object relative to its original position (in local space).

Get Object API

To perform any operation with an object, you first need to get it (i.e. receive the link to it). There are several API functions for doing this. A link to an object has `Object3D` type.

`get_object_by_name()` Get object by name.

```
// ...
var obj = m_scenes.get_object_by_name("Object");
// ...
```

`get_object_by_dupli_name()` Get the duplicated object by empty name and dupli name.

```
// ...
var obj = m_scenes.get_object_by_dupli_name("Empty", "Object");
// ...
```

`get_object_by_dupli_name_list()` Get the duplicated object by empty name and dupli name list (an array of `String` type elements).

```
// ...
var obj = m_scenes.get_object_by_dupli_name_list(["Empty1", "Empty2", "Object"]);
// ...
```

`get_object_name_hierarchy()` Returns the object names hierarchy array (from the highest parent to the object itself) for a given object.

```
// ...
var names = m_scenes.get_object_name_hierarchy(obj);
// ...
```

`check_object_by_name()` Check if object with given name is present on scene.

```
// ...
var object_exists = m_scenes.check_object_by_name("Cube");
// ...
```

`check_object_by_dupli_name()` Check if duplicated object is present on scene by empty name and dupli name.

```
// ...
var object_exists = m_scenes.check_object_by_dupli_name("Empty", "Cube");
// ...
```

`check_object_by_dupli_name_list()` Check if duplicated object is present on scene by empty name and dupli name list (an array of String type elements).

```
// ...
var object_exists = m_scenes.check_object_by_dupli_name_list(["Empty1", "Empty2", "Object"]);
// ...
```

`get_object_name()` Get the object's name.

```
// ...
var object_name = m_scenes.get_object_name(obj);
// ...
```

Object Selection

In order to enable selection of a certain object, it is required to enable the Selectable checkbox on the Selection and Outlining panel.

Note: Make sure that the status on the Scene > Object Outlining panel is set to ON or AUTO.

Object selection is possible programmatically via API, for example, in the scenes.js module there is the `pick_object` function which selects an object based on canvas 2D coordinates,

```
// ...
var x = event.clientX;
var y = event.clientY;

var obj = m_scenes.pick_object(x, y);
// ...
```

or using the Logic Editor.

If the selectable object has enabled Enable Outlining and Outline on Select checkboxes on the Object > Selection and Outlining panel, then the `pick_object` function call will activate outline glow animation.

Note: If the selected object is transparent (Blend, Add and Sort transparency types), outline glow will only be visible on the parts that have Alpha value higher than 0.5.

Copying Objects (Instancing)

It is often required to copy (to make instances of) objects during application work.

Copying objects has its limitations:

- only MESH objects can be copied
- the object should be **dynamic** (enable [Rendering Properties > Force Dynamic Object](#))
- the source object should belong to the active scene

Making a Simple Copy

In case of simple copying the new object will share the mesh with the original object. Thus, if the original object's mesh is changed, the copied object's mesh will be changed too. To make simple copying possible, it's enough to turn on the [Blend4Web > Force Dynamic Object](#) setting in the source object's properties.

Making a Deep Copy

In case of deep copying, the new object will have unique properties, namely it will have its own mesh. Thus, if the original object's mesh is changed, the copied object's mesh will not be changed. Also, the canvas textures on the copied objects are different textures and not one and the same like it is the case with the simple copying. To make deep copying possible, it is required to enable the [Rendering Properties > Dynamic Geometry](#) checkbox for the source object. |

Copying objects in runtime can be performed with the `copy` method of the `objects.js` module. This method requires three arguments: the id of the source object, a unique name for the new object and the boolean value to specify the copy mode (i.e. simple or deep). By default, simple copying will be performed.

The newly created object should be added to the scene. This can be performed with the `append_object` method of the `scenes.js` module. The new object should be passed to it as an argument.

```
// ...
var new_obj = m_objects.copy(obj, "New_name", true);
m_scenes.append_object(new_obj);
m_transform.set_translation(new_obj, 2, 0, 2);
// ...
```

Removing Objects

To remove objects, use the `remove_object` method of the `scenes.js` module. Pass the object to it as an argument. Dynamic mesh- and empty-type objects can be removed

this way.

```
// ...
m_objects.remove_object(new_obj);
// ...
```

Quaternions

Quaternion is a four-component vector used to perform rotating. Quaternions have a number of advantages over other rotation methods such as:

- A quaternion has no ambiguity and doesn't depend on the rotation order as the Euler angles.
- Quaternion's memory usage is more effective (2-4 times less depending on the matrix used).
- Better computing efficiency than for matrices in case of a series of rotations.
- Numeric stability - compensation for multiplication errors arising from float number inaccuracy.
- Convenient interpolation method.

Quaternions have some drawbacks:

- Rotating a vector with a quaternion is more computationally expensive than rotating with a matrix.
- It is difficult to use quaternions for non-rotation transformations (such as perspective or orthogonal projection).

The engine has a number of functions to make it more convenient to work with quaternions:

`quat.multiply` Quaternion multiplication. Note that left-multiplying A quaternion by B quaternion A*B is a rotation by A. I.e. the object already has some rotation B which we supplement with a new rotation by A.

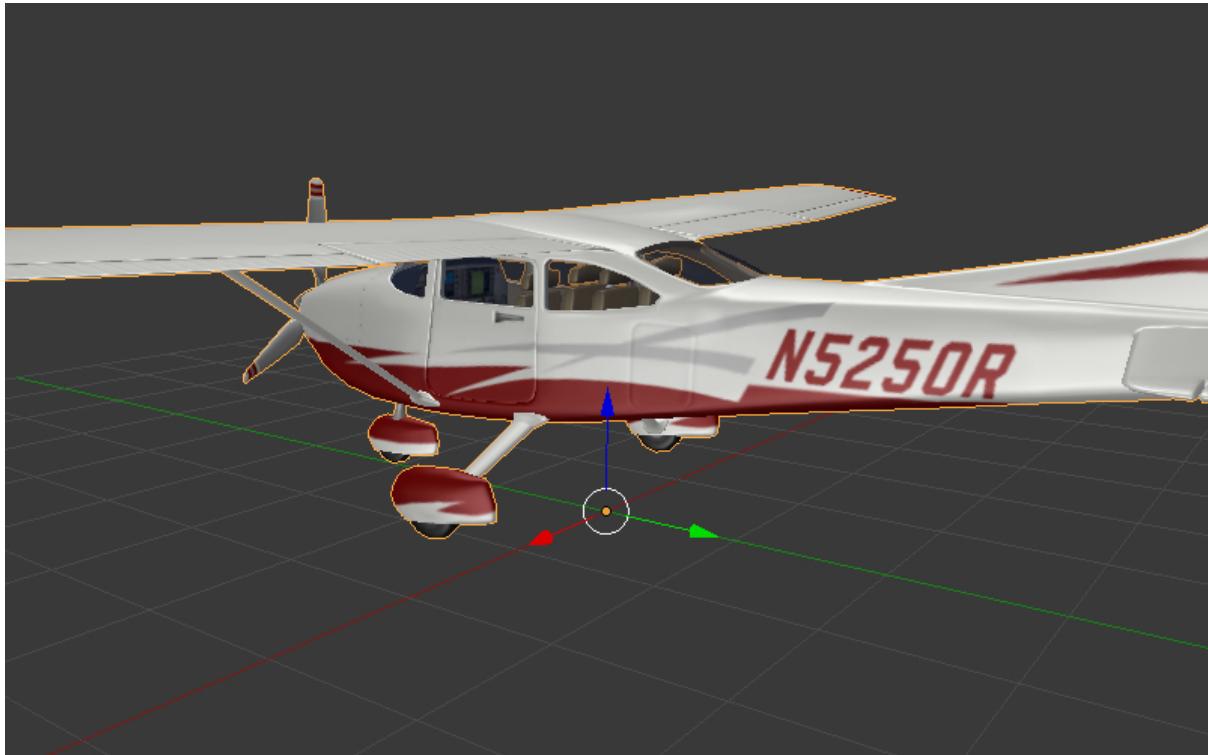
`quat.setAxisAngle` A quaternion is an alternative presentation of rotation by an arbitrary angle relative to the arbitrary axis (vector). Positive direction of rotation is defined as anticlockwise when viewing from the vector's end. For example the `'quat.setAxisAngle([1, 0, 0], Math.PI/2, quat)'` call forms a quaternion which can be used for rotating the object by 90 degrees (anticlockwise if viewing from the X axis' end) relative to the X axis.

`quat.slerp` Spherical interpolation of quaternions. Used for smoothing the object's rotation and animation.

`util.euler_to_quat`, `util.quat_to_euler`. Conversion from Euler angles and back.

Quaternion Example

We need to rotate the object by 60 degrees in a horizontal plane to the right. We have a model named “Cessna” in Blender.



Let's save a reference to the object in the `aircraft` variable:

```
var aircraft = m_scenes.get_object_by_name("Cessna");
```

Let's rotate it:

- The orientation of coordinate axes is different in Blender and in the engine. Upon export there will be a transformation [X Y Z] (Blender) -> [X -Z Y] (the engine). Therefore we need to rotate the object relative to the Y axis and not the Z axis.
- A clockwise rotation corresponds to the rotation to the right (i.e. in the negative direction).
- 60 degrees = $\pi/3$ radians.

As a result we get:

```
// compose quaternion
var quat_60_Y_neg = m_quat.setAxisAngle([0, 1, 0], -Math.PI/3, m_quat.create());

// get old rotation
var quat_old = m_transform.get_rotation(aircraft);
```

```
// left multiply: quat60_Y_neg * quat_old
var quat_new = m_quat.multiply(quat_60_Y_neg, quat_old, m_quat.create());

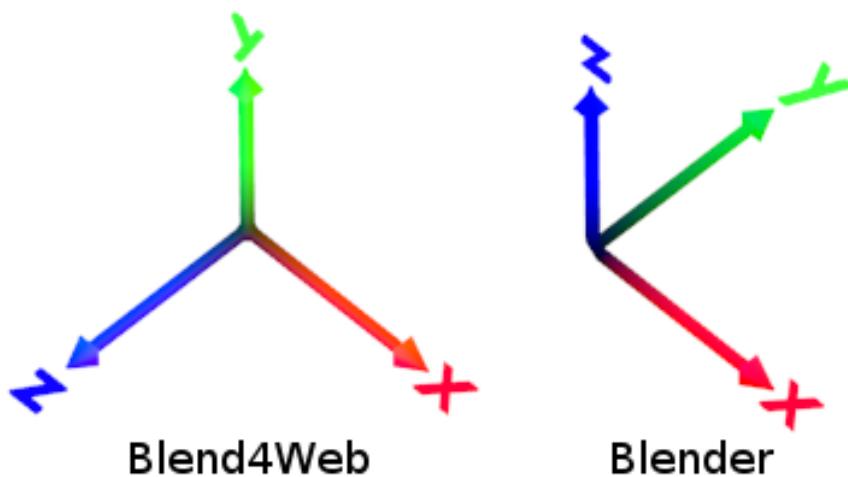
// set new rotation
m_transform.set_rotation_v(aircraft, quat_new);
```

The optimized version which does not create new objects:

```
// cache arrays as global vars
var AXIS_Y = new Float32Array([0, 1, 0]);
var quat_tmp = new Float32Array(4);
var quat_tmp2 = new Float32Array(4);
...
// rotate
m_quat.setAxisAngle(AXIS_Y, -Math.PI/3, quat_tmp);
m_transform.get_rotation(aircraft, quat_tmp2);
m_quat.multiply(quat_tmp, quat_tmp2, quat_tmp);
m_transform.set_rotation_v(aircraft, quat_tmp);
```

Differences Between Coordinate Systems of Blender and Blend4Web

In Blender's coordinate system the UP vector, which points upwards, is co-directional with the Z axis. Blend4Web uses Y axis for this purpose, as it is customary in OpenGL. Thus the engine's coordinates are rotated by 90° around the X axis relative to Blender.



API methods use the engine's coordinates, so they can work differently in comparison

with setting Blender's parameters.

Moving via TSR Vectors

It is sometimes convenient to move objects using vectors of the following format:

$$[T_x, T_y, T_z, S, R_x, R_y, R_z, R_w]$$

Here T_x, T_y, T_z - the components of the translation vector, S - scale factor, R_x, R_y, R_z, R_w - the components of the quaternion vector. Hence the name of this vector: TSR or TSR-8.

This vector can be operated via `tsr` module, as well as via `set_tsr()`/`get_tsr()` methods of the `transform` module.

Meshes

Table of Content

- Meshes
 - Static and Dynamic Meshes
 - Settings
 - * Override Bounding Volumes
 - Morphing

Meshes are a subclass of [objects](#). Meshes are different from objects of the other types in a sense that they have geometric structure that is visible in the scene. All objects user can see in a scene are either meshes or converted to meshes on export (like the CURVE, TEXT, METABALL and SURFACE type objects).

Blend4Web addon has several tools for mesh editing, including the [normal editor](#).

This chapter provides an overview of the MESH type object parameters and settings, as well as the API functions to work with them.

Static and Dynamic Meshes

All MESH objects can be divided into static and dynamic.

Static meshes are meshes that can be merged together if they have the same material.

Dynamic meshes are meshes that cannot be combined with each other.

Merging of static meshes - so called batching - is performed automatically when the scene is loaded in order to optimize the number of draw calls. The conversion is performed even if there is just one object in the scene. The center of the resulting mesh is located in the origin.

Meshes that have animation, physics or a parent, which is a dynamic object, are considered dynamic.

Settings

Meshes have all settings of the [objects](#) and several additional settings.

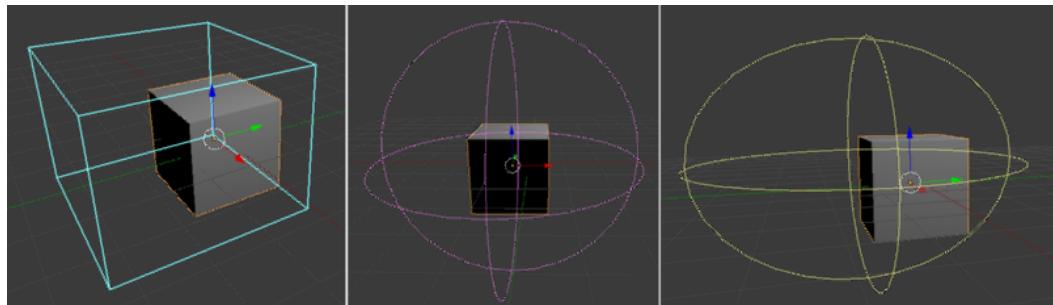
Override Bounding Volumes

This section is used to override bounding volumes. Such volumes are used to check object's visibility, to handle collisions and physics interaction. By editing them, you can achieve various effects.



The Override Bounding Volumes section can be found on the Blend4Web panel (beside the [normals editor](#).).

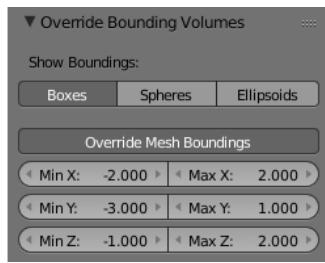
Show Boundings



Show the bounding volumes in the Blender viewport. This option works only if the Override Mesh Boundings parameter is enabled. The volume can be rendered as a rectangular cuboid or as a combination of circles or ellipses. More than one method of rendering can be active at the same time (even all three methods can be enabled simultaneously).

Note: The selected rendering method does not affect the bounding volume shape, only the way it is represented in the viewport.

Override Mesh Boundings



This button enables the override of the basic mesh boundings. The bounding volume always has the shape of a rectangular cuboid with a center in the object's pivot point.

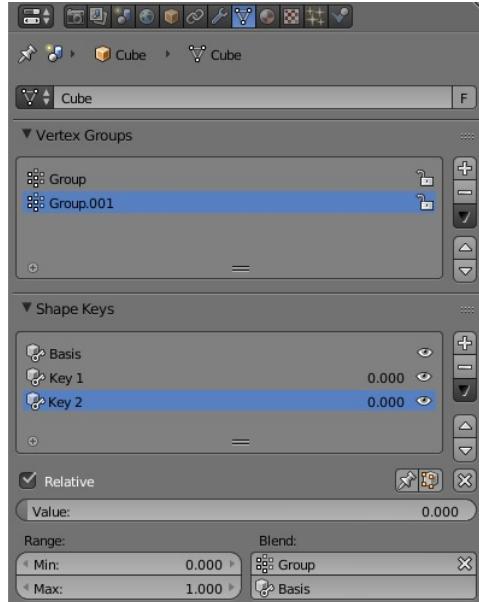
MinX and MaxX The volume's X dimensions. By default, MinX = -1, MaxX = 1

MinY and MaxY The volume's Y dimensions. By default, MinY = -1, MaxY = 1.

MinZ and MaxZ The volume's Z dimensions. By default, MinZ = -1, MaxZ = 1

Morphing

Morph targets can be added using Blender's standard Mesh > Shape keys interface.



The engine supports all shape key options under the "Relative" type.

To set a shape key value, use the `apply_shape_key` method of the `geometry.js` module.

Note: The object must have Export Shape Keys parameter enabled.

```
// ...
var obj = m_scenes.get_object_by_name("Object");
m_geometry.apply_shape_key(obj, "Key 1", 0.5);
// ...
```

Normal Editor

Table of Contents

- Normal Editor
 - Main Features of the Normal Editor
 - Interface of the Vertex Normal Editor
 - Activate
 - Show Normals
 - Rotation
 - Split Normals
 - Average Split
 - Restore
 - Tree, Foliage and Face
 - Copy/Paste
 - Copy From Mesh

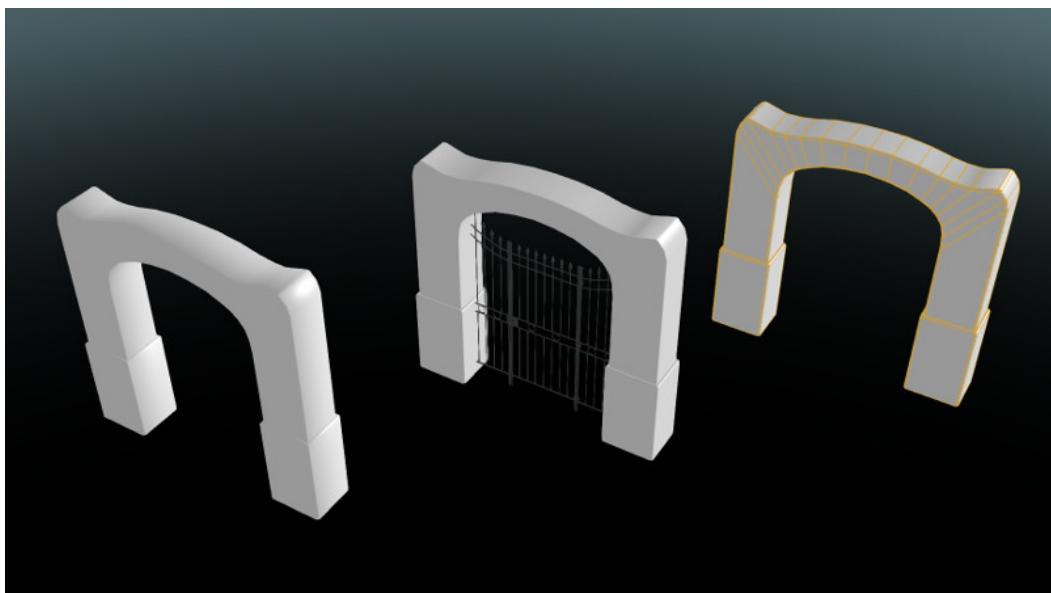
Editing the selected normals is a pretty easy and effective way to customize 3d model shading without complicating its geometry.

In some cases, using the normal editor you may achieve the effect which is similar to the result of using [normal maps](#). At the same time, editing normals is preferred because it is more computationally effective and consumes less video memory.

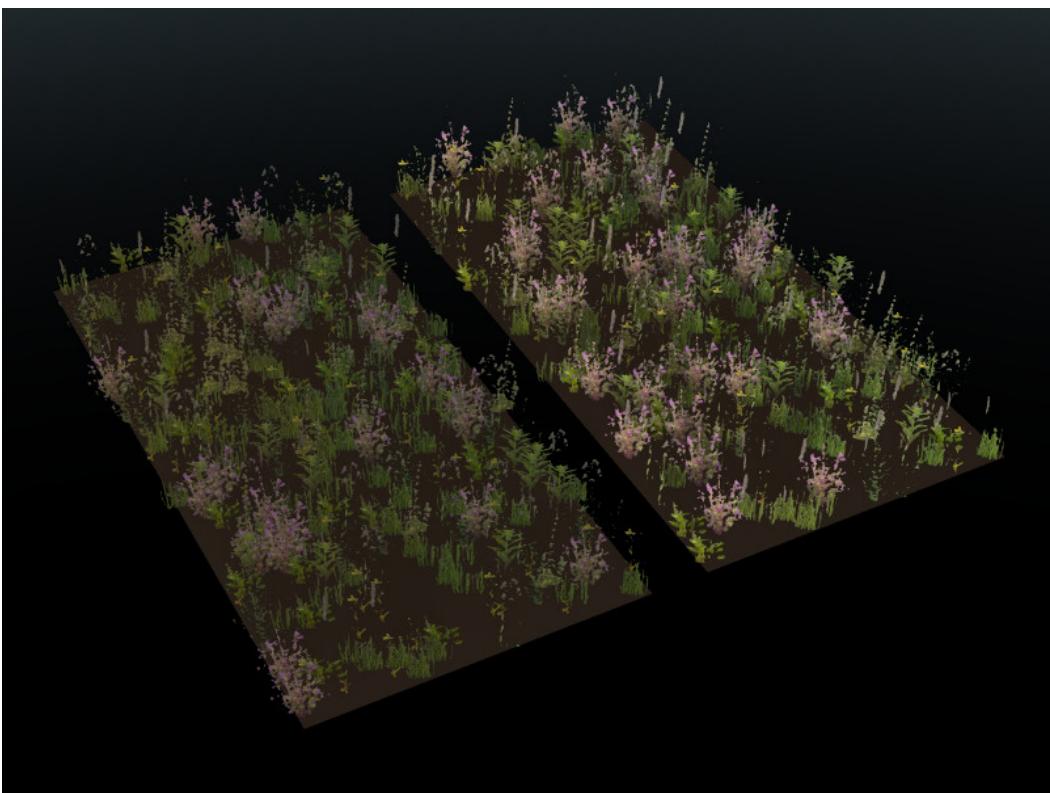
Normal editor workflow example:



Simple geometry shading is to the left, a tree with edited normals is to the right.



To the left - gates with common shading; at the center - gates with edited normals; to the right - geometry with a wireframe.



To the left - common grass geometry shading; to the right - grass with edited normals.



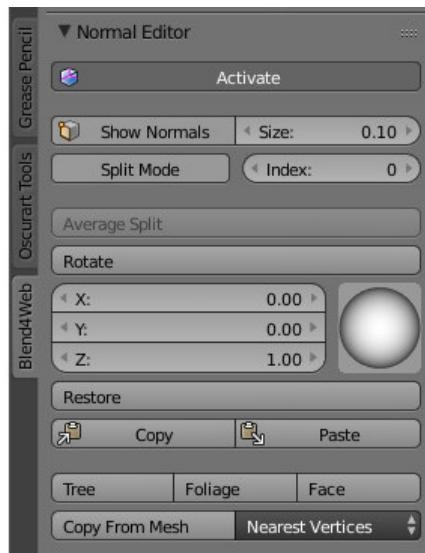
To the left - glasses with common shading; to the right - geometry with edited normals shading.

Main Features of the Normal Editor

1. native Blender storage is used as a container for edited vertices normals directions (it appeared in Blender 2.74);
2. normals editing and shading visualization are processed in Edit Mode now;
3. all changes are being saved automatically;
4. selected vertex normal rotation can be performed directly in the Viewport window with Shift+Ctrl+R hotkey, similarly to other rotation operations in Blender;
5. edited normals are being exported automatically.

Interface of the Vertex Normal Editor

The interface of the normal editor is located on the tools panel Blend4Web > Normal Editor. The Shading: Smooth mode should be enabled and Activate button should be pressed or Auto Smooth flag should be enabled in the mesh settings, before starting to work with the editor.



Activate

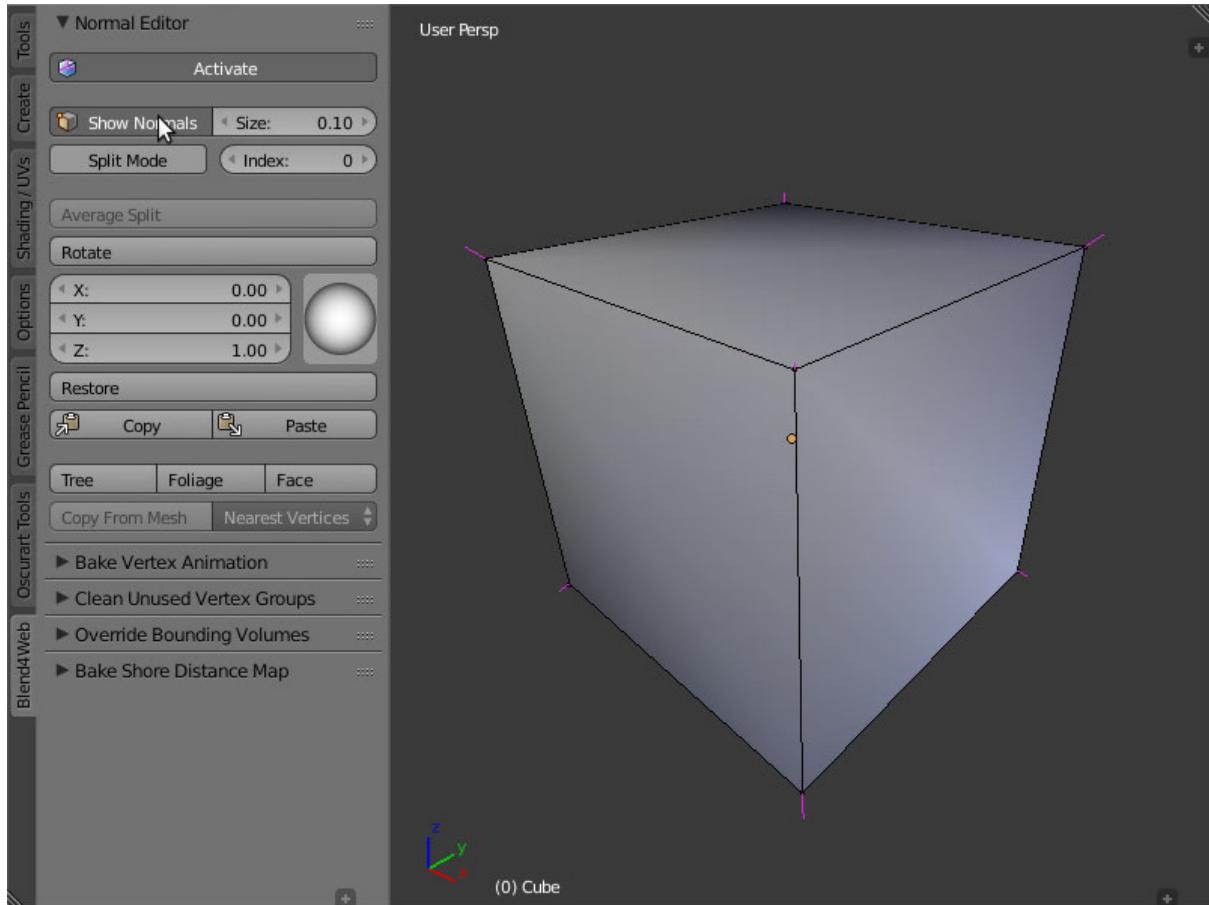
The Activate button turns on vertex normal editing mode.

You can just press Activate button while in Edit Mode and start editing vertex normals. As it is active, object shading and its export would be processed taking edited vertex normals into account. In other words, after making some changes, this button should be left active if you want to see the changes in the Blender Viewport and in the Blend4Web engine.

Show Normals

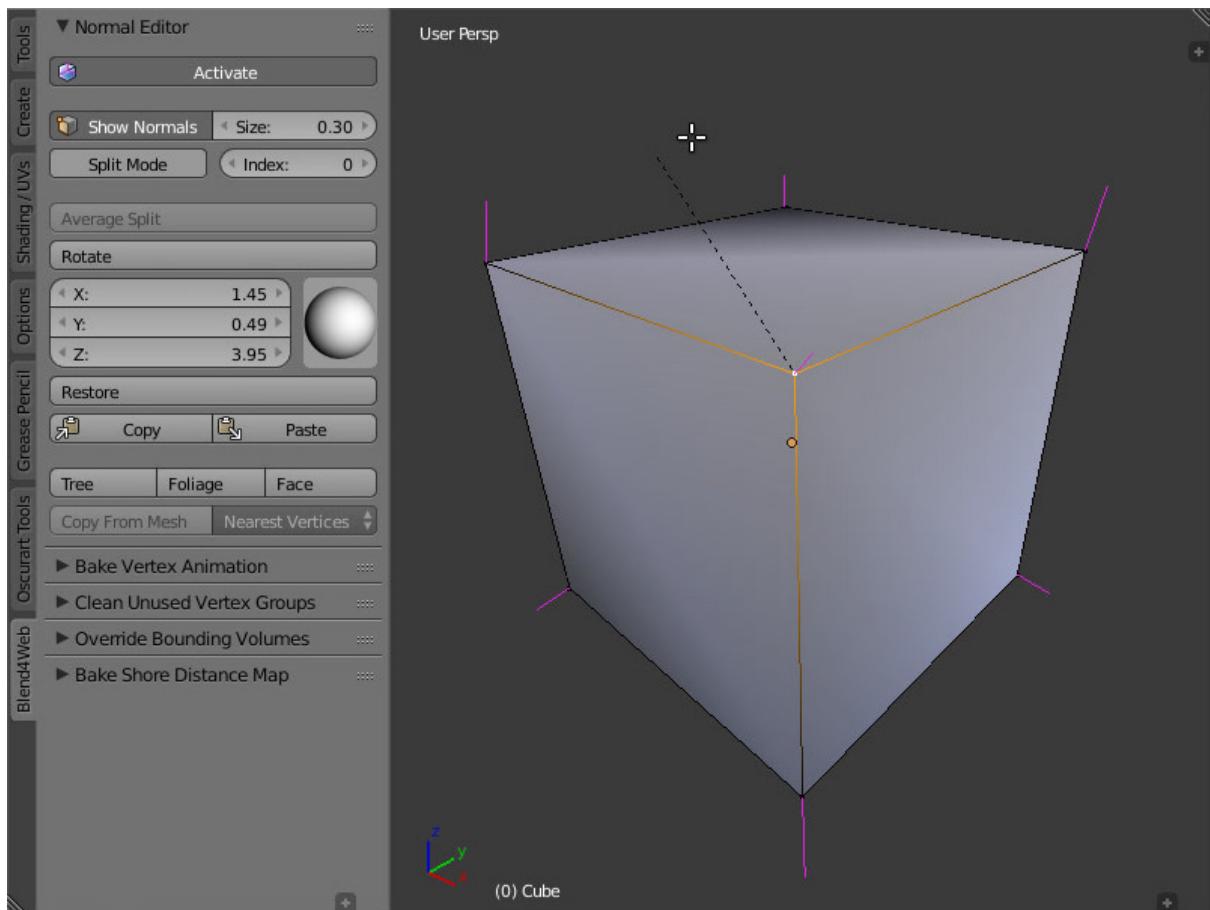
This button actually duplicates the original Blender's button. It turns on displaying the normals in the viewport, while the Size field allows you to set their length.

You just need to push the Show Normals button either on the normal editor panel or on the Blender's right panel in the Mesh Display section. You can also set the convenient length of normals by tweaking the Size.



Rotation

Using these instruments you can change direction of the normals. The Rotate function is also available through the Shift+Ctrl+R hotkeys, which allows rotation of vertex normals similarly to Blender.

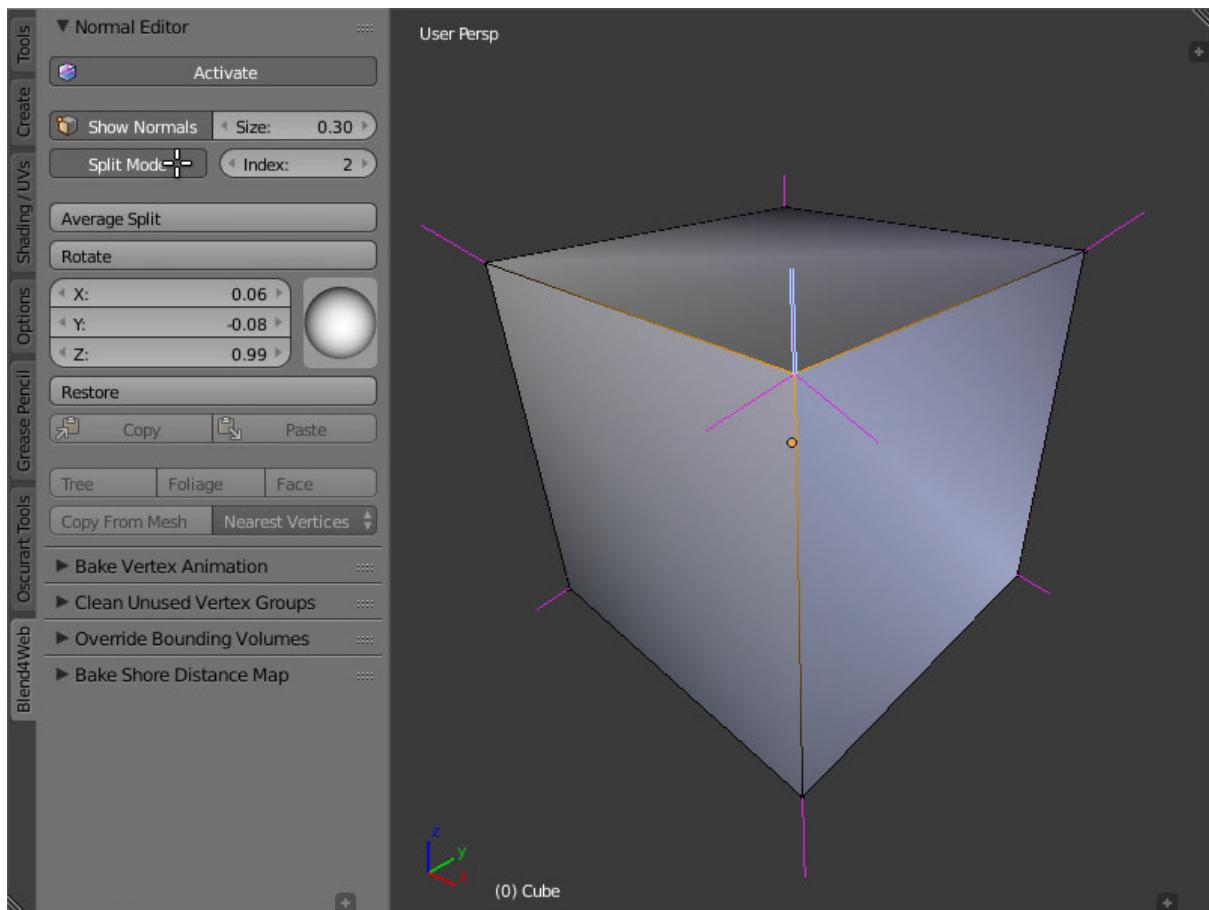


Select one or more vertices that you want to edit and then rotate their normals using the visual sphere or specify the direction as numerical values.

The Rotate button provides a more convenient way to manipulate normals. The rotation is performed in the screen space. Nevertheless, as with any other rotations in Blender, you can isolate the desired axis during rotation (by typing X, Y or Z).

Split Normals

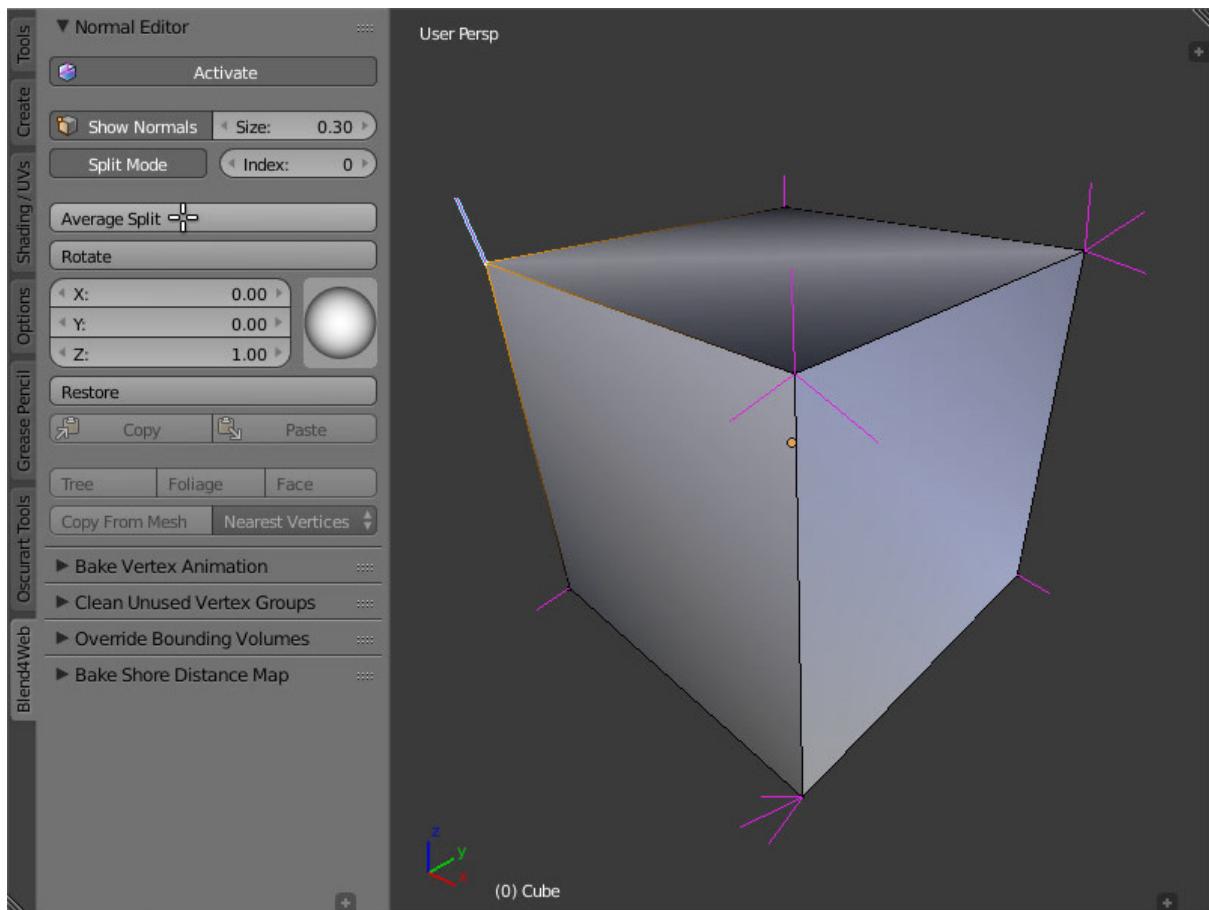
The Split Normals mode allows to edit vertex normals separately for each face that form the vertex. Index allows you to navigate between the split normals.



Turn on the Split Normals mode, select the vertex and change the direction of its normals. Firstly, the normal which has the zero index in the queue will be modified. Then, by switching between indexes you can go to the next normal of this vertex and edit it, then to the next and so on.

Average Split

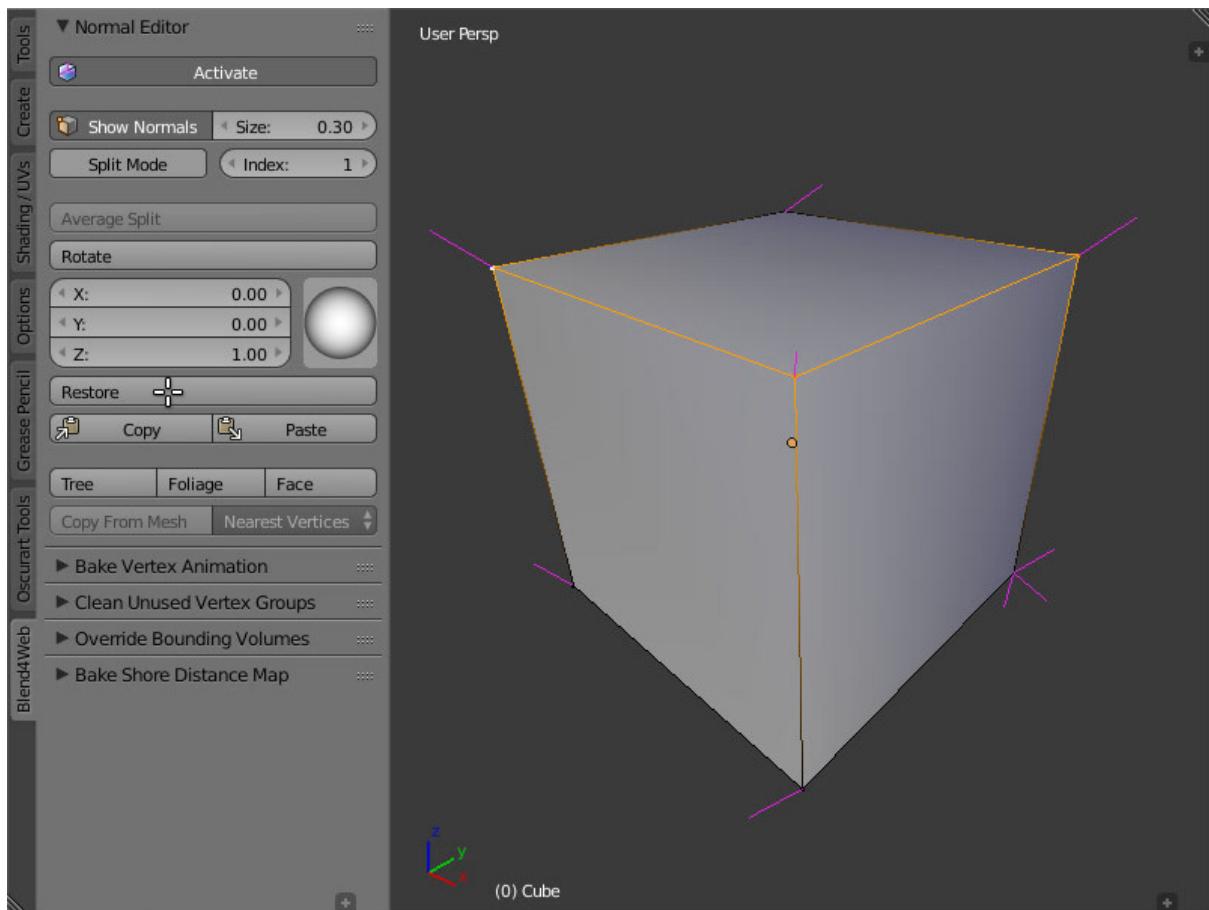
The Average Split button averages the direction of the vertex normals which was modified in the Split Normals mode.



To combine several split vertex normals into one, in order to obtain the average direction of these normals, you just need to select the desired vertex and press the Average Split button.

Restore

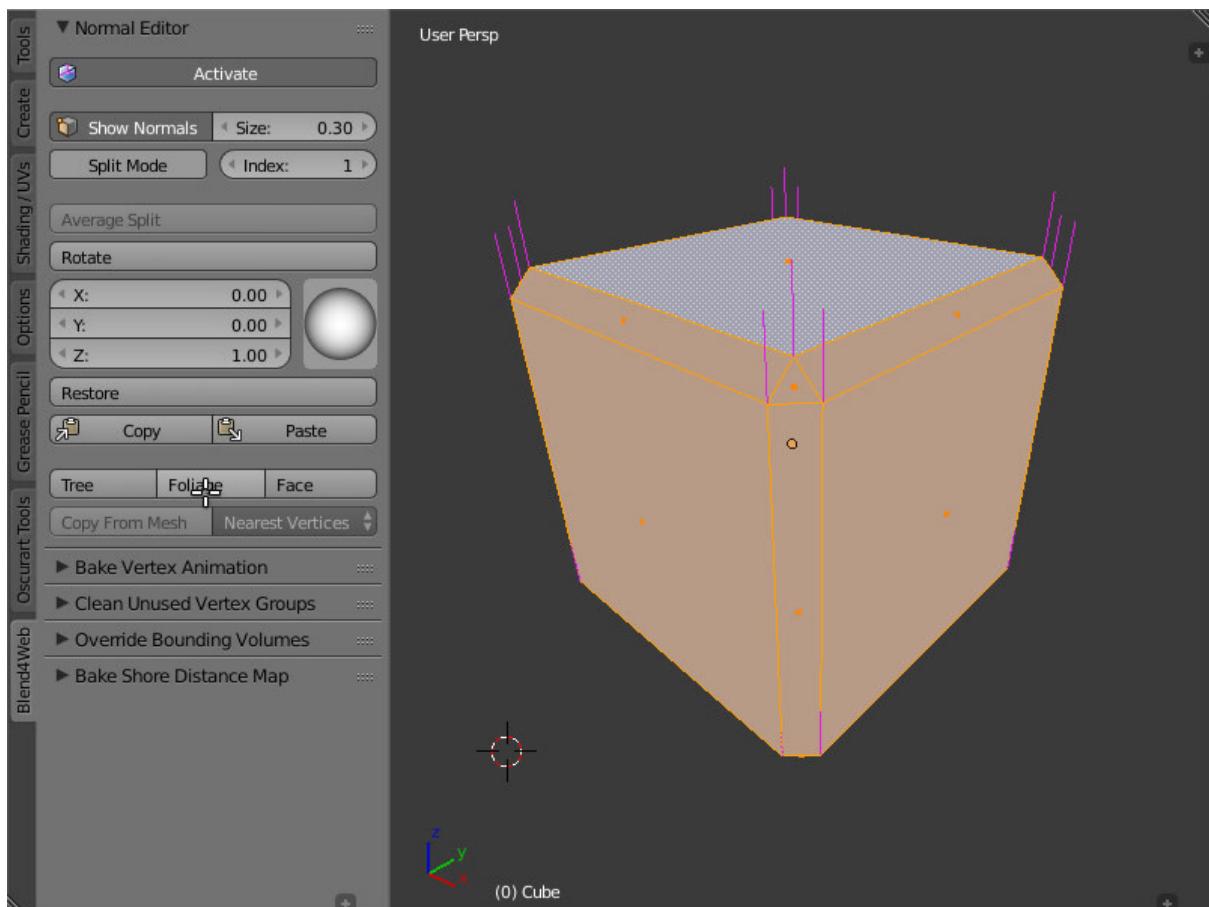
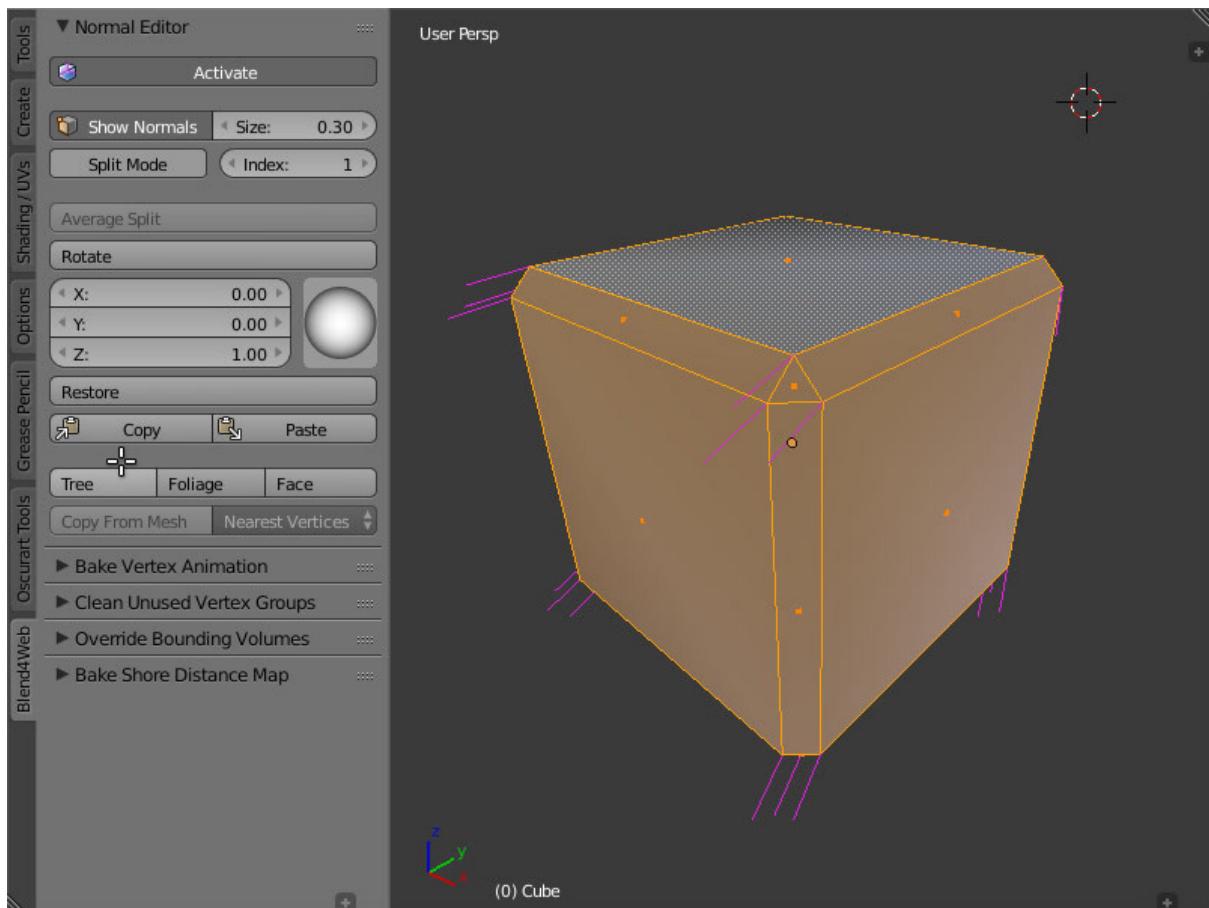
The Restore button restores the original direction of normals for the selected vertices.

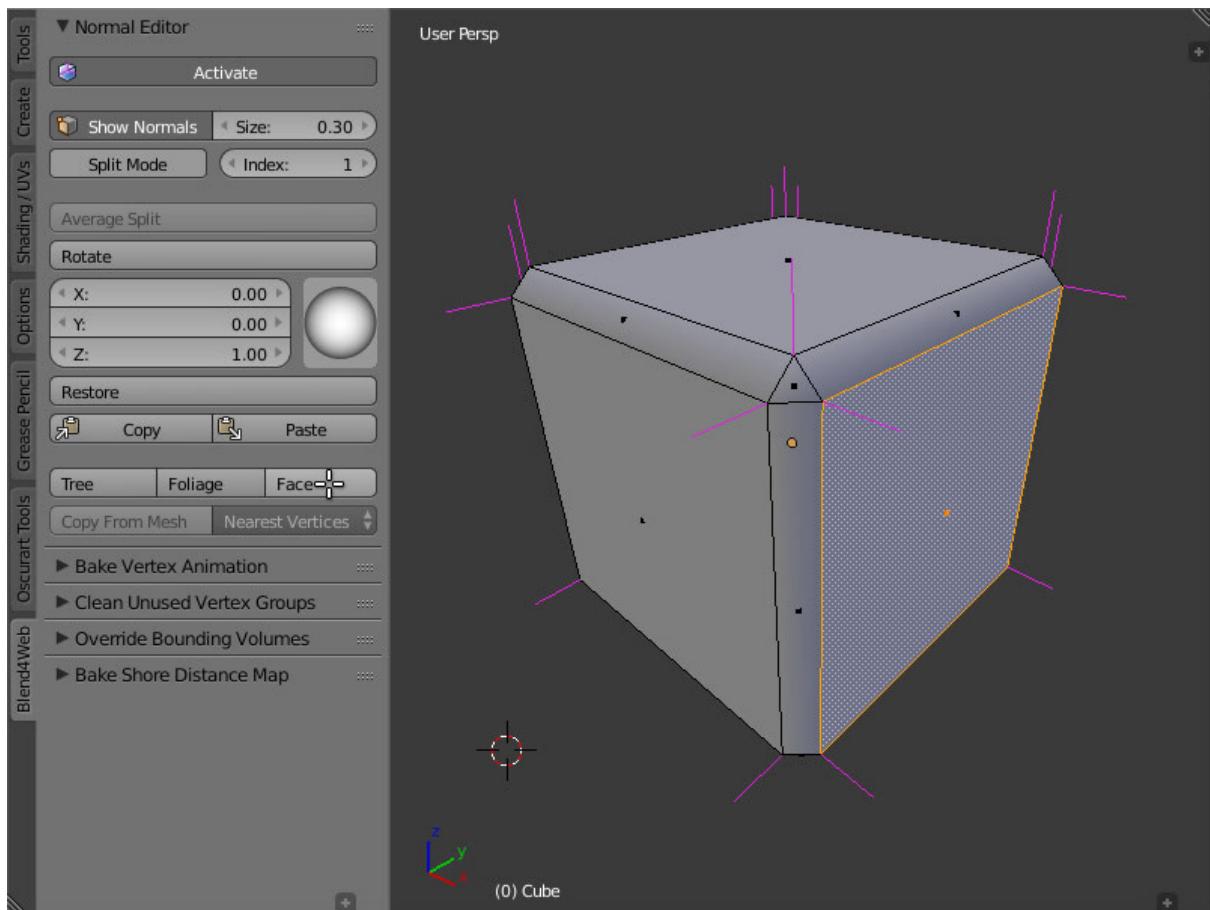


In order to restore the normal's direction to its original (which is calculated on the basis of face normals), you need to select the desired vertices and click the **Restore** button.

Tree, Foliage and Face

The **Tree** button directs the normals of the selected vertices away from the 3D cursor. The **Foliage** button directs the normals of the vertices strictly up. The **Face** button directs the normals of the selected face parallel to the normal of this face.





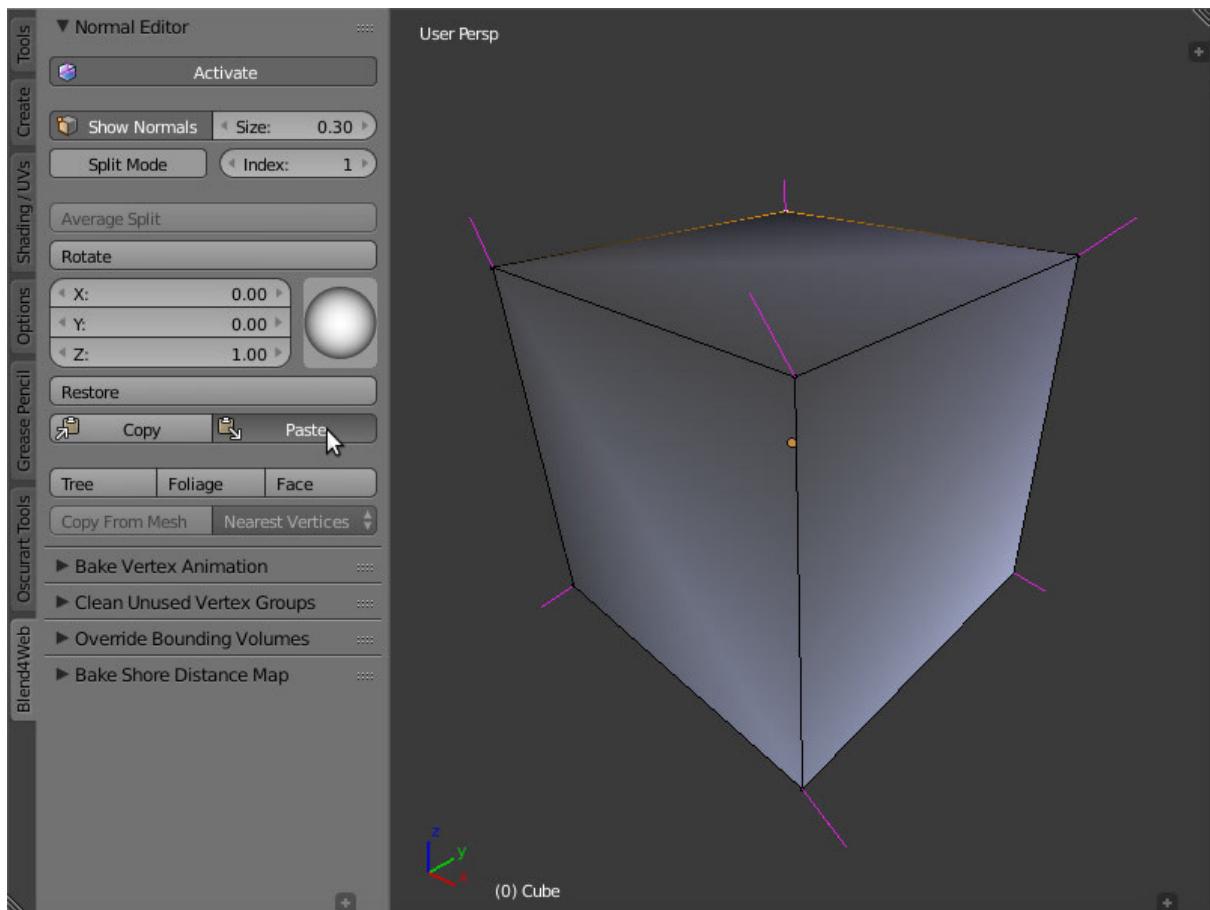
In order to use the Tree function, select the desired vertices and place the 3D cursor in the desired position. Then click the Tree button so all the selected vertices will turn their direction away from the cursor, as if they were shot from one point.

The Foliage function is very easy to use: just select the vertices and press the button, so the function will direct their normals straight up.

In order to direct the normals parallel to the face normal, just select the desired face and click the Face button. The normals of the vertices which form the face will be directed parallel to the face normal. This function works only with one selected face at a time.

Copy/Paste

Copies the normal direction from one vertex to another.

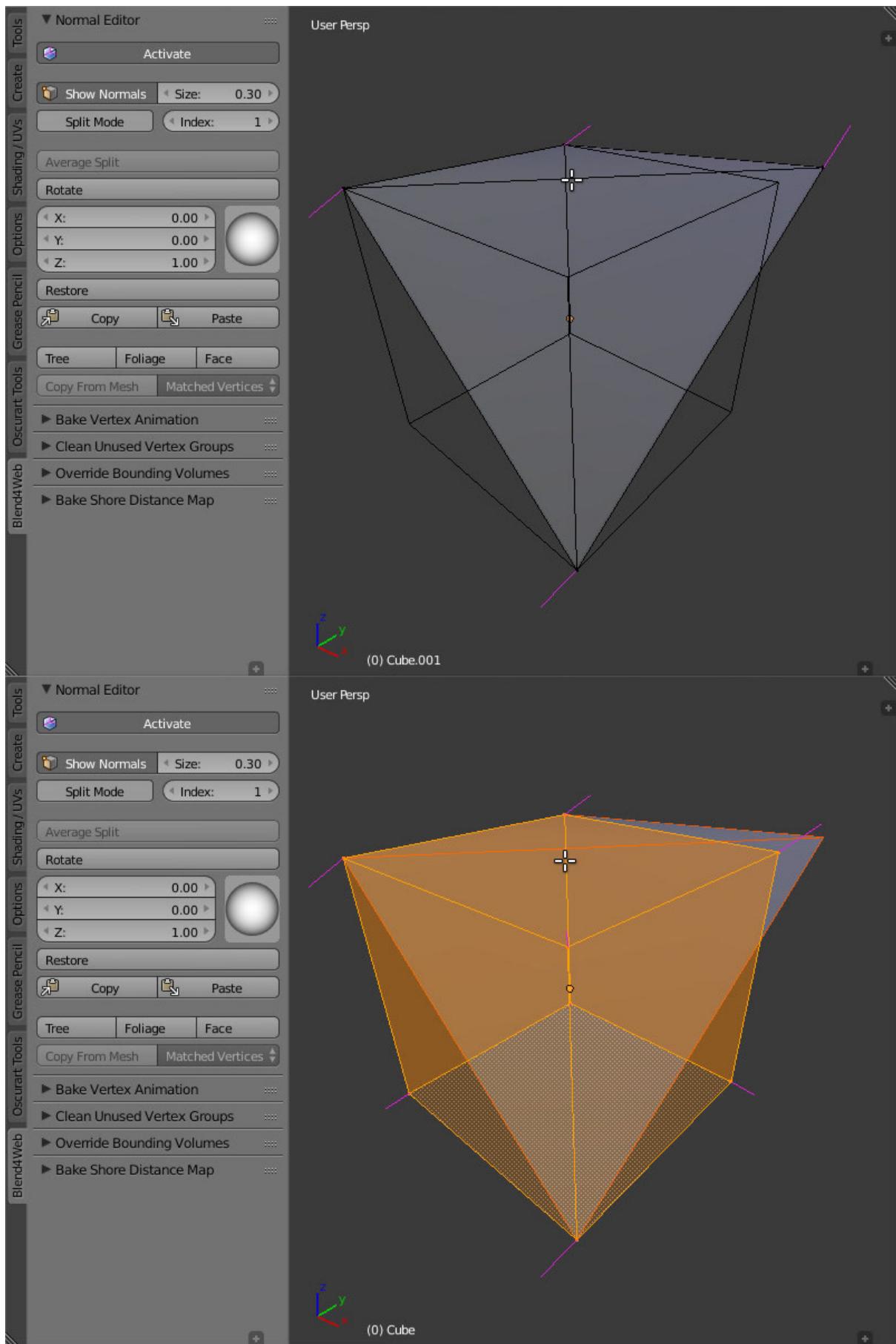


Select the vertex you want to copy from and click the Copy button. Then, select the vertex you want to copy to and click the Paste button. You may copy information from one selected vertex to many different vertices. The buttons are not active in the Split Mode and do not copy data of split vertices.

Copy From Mesh

This function allows you to copy the normals from one object to another. There are two modes: Matched Vertices and Nearest Vertices.

The Matched Vertices mode copies the normals from the vertices of one object to another object's vertices only if they have the same coordinates; the other vertices are ignored. The Nearest Vertices mode copies the normals of the nearest vertices of the source object.



This function works in Blender's object mode. At first, you need to select the object you need to copy from, then the object you need to copy to. It is also necessary to select the target object's vertices to which normals are copied.

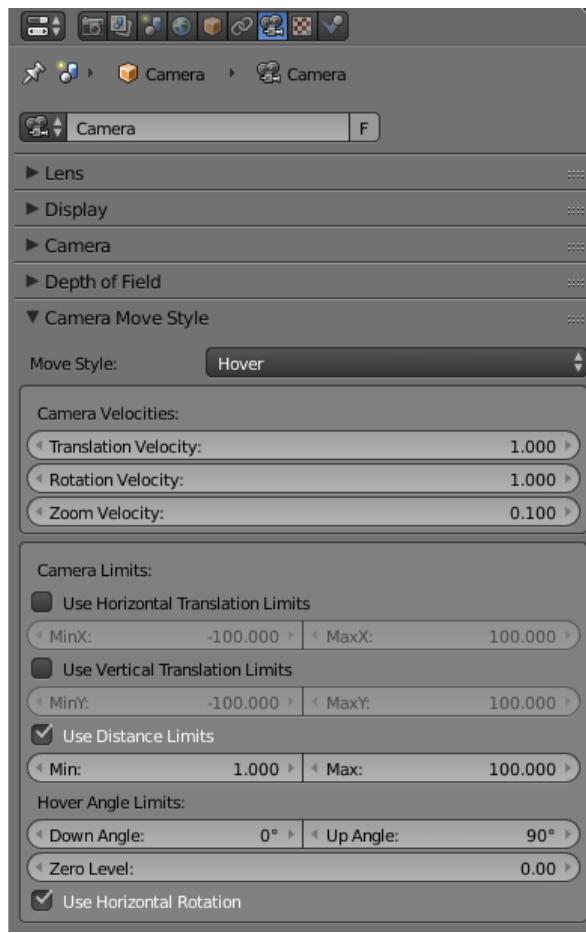
Camera

Table of Contents

- Camera
 - Move Styles and General Settings
 - Limiting the Camera Movement
 - * Target Type Camera
 - * Eye Type Camera
 - * Hover Type Camera
 - Viewport Alignment
 - Camera Controls API
 - * Control Mode Setup
 - * Translation and Rotation
 - * Limit Setup
 - * Angular Coordinates Count

Move Styles and General Settings

The camera settings are specified in the Properties panel under the Camera (Object Data) tab.



Camera Move Style > Move Style Camera control mode:

- Target By default the camera is rotating around a fixed point (target). The pivot's position can be changed (see [camera panning](#)).
- Eye The Eye mode allows rotation and translation as in first person view.
- Hover In Hover mode, camera moves parallel to the horizontal plane. By using additional limits, a smooth movement path toward the pivot point for the camera can be created.
- Static In the Static mode the camera can be moved via animation or through API calls.

Look At Cursor > Target Location Available for the Target mode. This is the position of the camera pivot point. The Look At Cursor button copies the current 3D cursor position into this value.

Depth of Field Described in the [Depth of Field](#) section.

Velocity settings are available for camera movement.

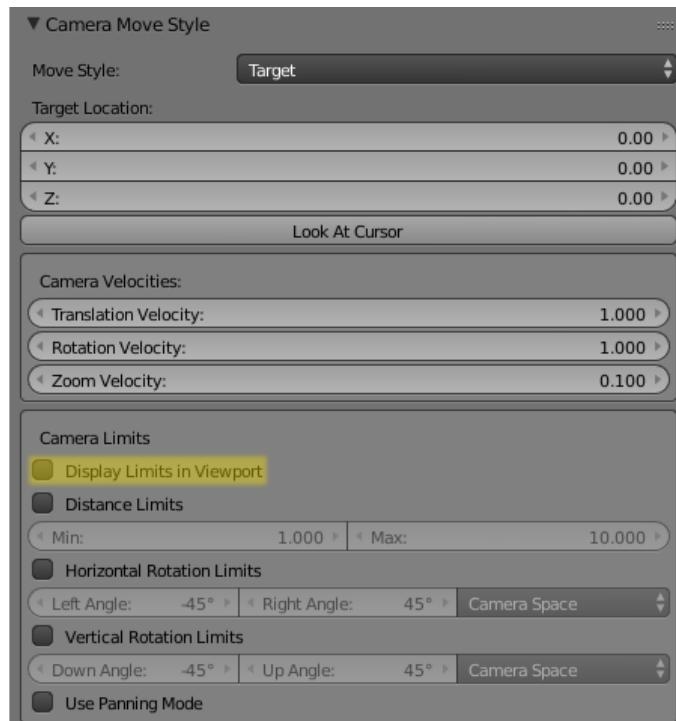
Camera Move Style > Camera Velocities > Translation Velocity Available for the Target, Eye and Hover type cameras. Sets the velocity of the camera translation. Possible value interval: $[0, \infty)$. Default value is 1.

Camera Move Style > Camera Velocities > Rotation Velocity Available for the Target, Eye and Hover type cameras. Sets the camera rotation speed. Possible value interval: $[0, \infty)$. Default value is 1.

Camera Move Style > Camera Velocities > Zoom Velocity Available for the Target and Hover type cameras. Set the velocity of the camera approaching the pivot point. Possible value interval: $[0, 0.99]$. Default value is 0.1.

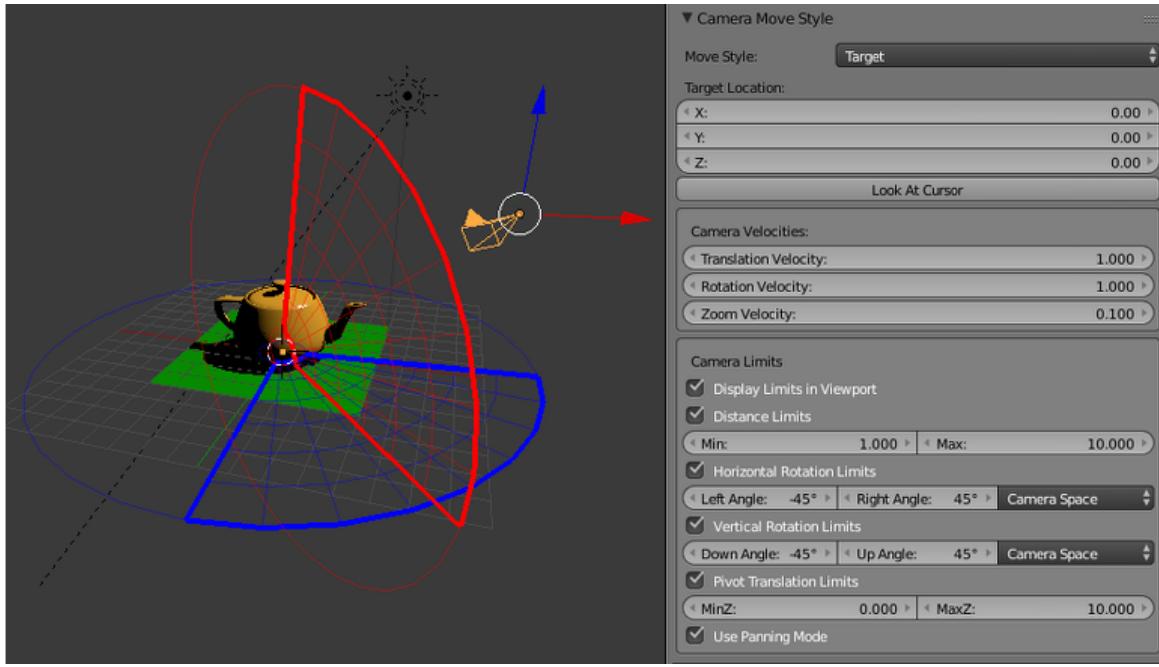
Limits the Camera Movement

There are several settings for the camera which limit/change its movement one way or another. They are grouped as Camera Limits.

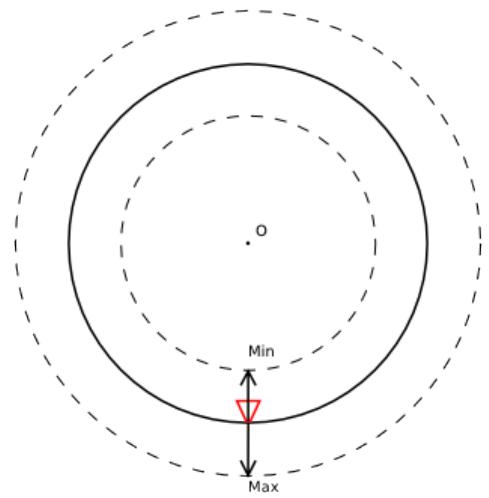


Camera limits can be viewed right in the Blender viewport, if the Display Limits in Viewport parameter is enabled.

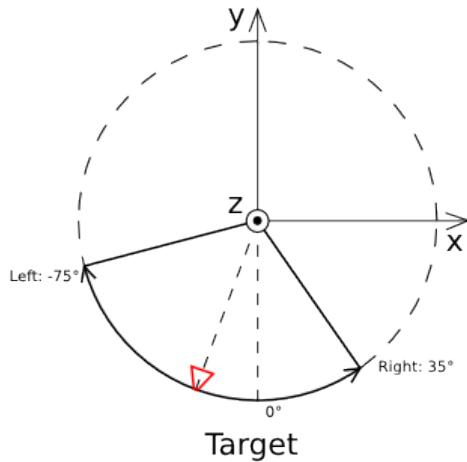
Target Type Camera



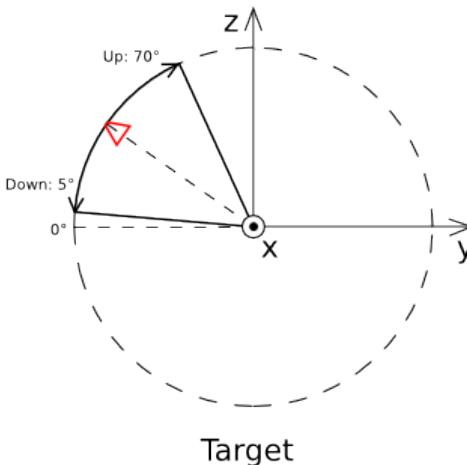
Camera Move Style > Camera Limits > Distance Limits Set minimum and maximum distances from the camera to the pivot point. Allowable values: $\text{Min} \leq \text{Max}$. Default values: $\text{Min} = 1$, $\text{Max} = 10$. Disabled by default.



Camera Move Style > Camera Limits > Horizontal Rotation Limits Limit the horizontal (around the Z world axis in Blender) camera rotation around the corresponding point. Rotation is performed along the arc of a circle between Left Angle and Right Angle values. The rotation arc corresponds to movement from Left Angle to Right Angle anticlockwise. Default values: Left Angle = -45° , Right Angle = 45° . Disabled by default.



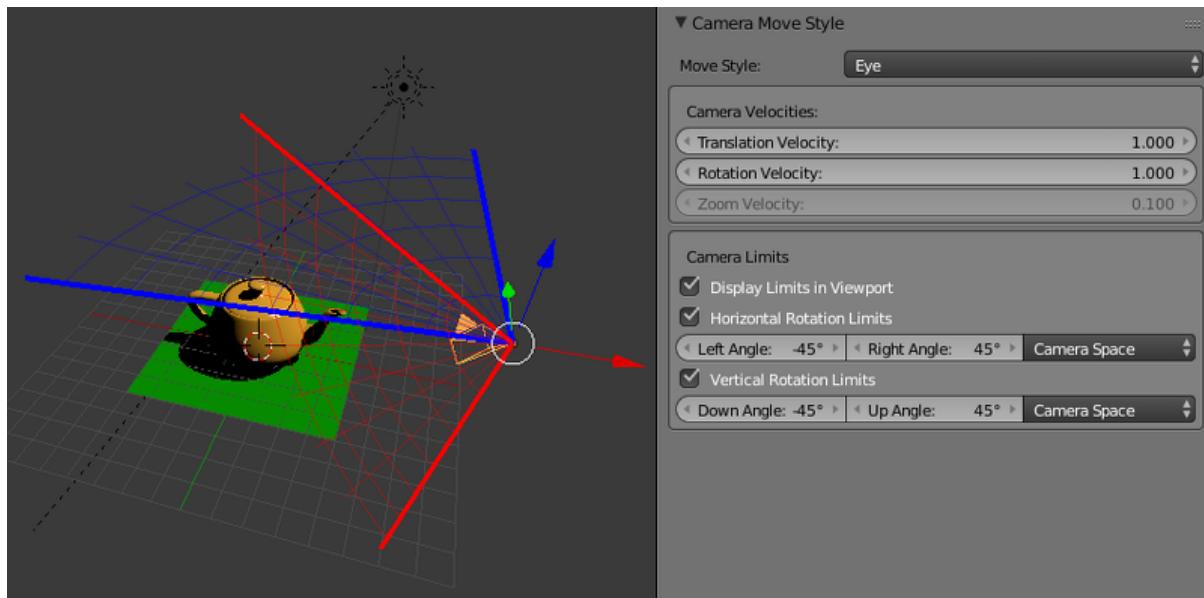
Camera Move Style > Camera Limits > Vertical Rotation Limits Limit the horizontal (around the Z world axis in Blender) camera rotation around the corresponding point. Rotation is performed along the arc of a circle between Left Angle and Right Angle values. The rotation arc corresponds to movement from Left Angle to Right Angle anticlockwise. Default values: Left Angle = -45° , Right Angle = 45° . Disabled by default.



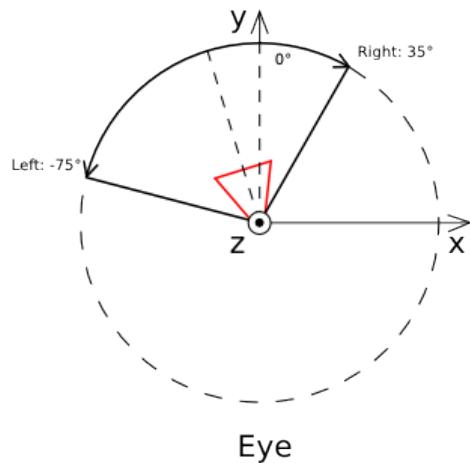
Camera Move Style > Camera Limits > Pivot Translation Limits Limit the translation of the camera's pivot point. Default values: MinZ = 0, MaxZ = 10. Not rendered in the viewport.

Camera Move Style > Camera Limits > Use Panning Mode Allow camera panning.

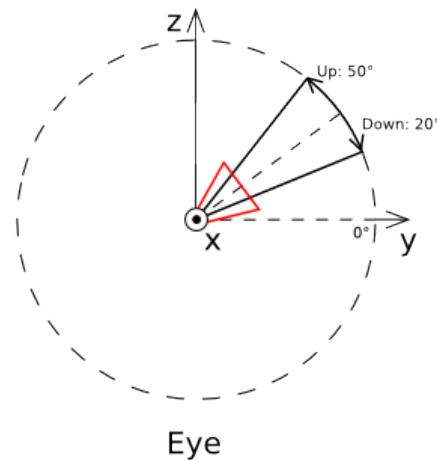
Eye Type Camera



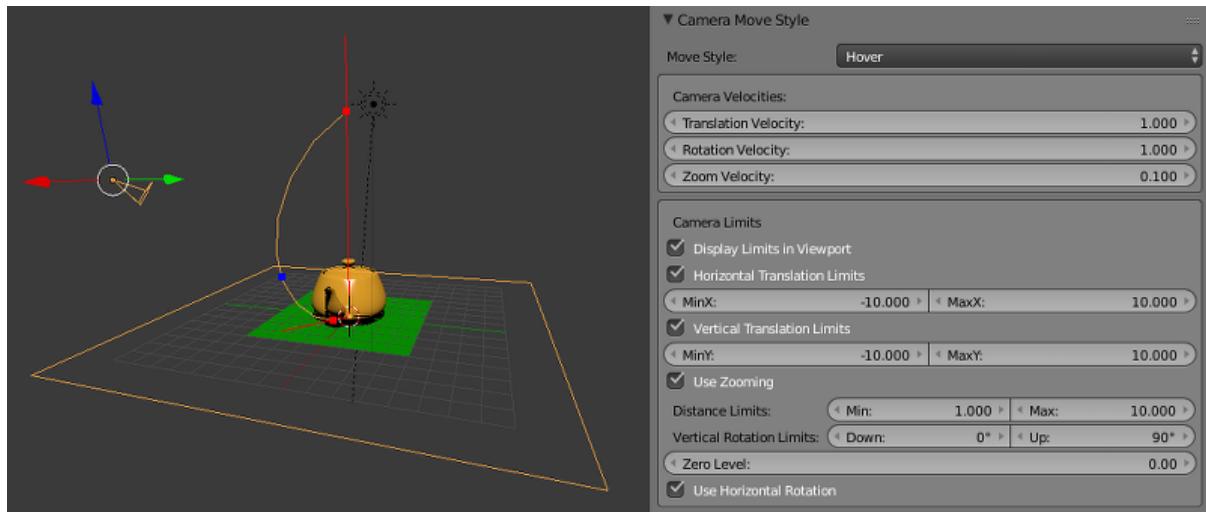
Blend4Web > Horizontal Rotation Limits Limit the horizontal (around the Z world axis in Blender) camera rotation around the corresponding point. Rotation is performed along the arc of a circle between Left Angle and Right Angle values. The rotation arc corresponds to movement from Left Angle to Right Angle anticlockwise. Default values: Left Angle = -45° , Right Angle = 45° . Disabled by default.



Camera Move Style > Camera Limits > Vertical Rotation Limits Limit the vertical (around the local X axis in Blender) camera rotation around the corresponding point. Rotation is performed along the arc of a circle between Down Angle and Up Angle values. The rotation arc corresponds to movement from Down Angle to Up Angle anticlockwise. Default values: Down Angle = -45° , Up Angle = 45° . Disabled by default.

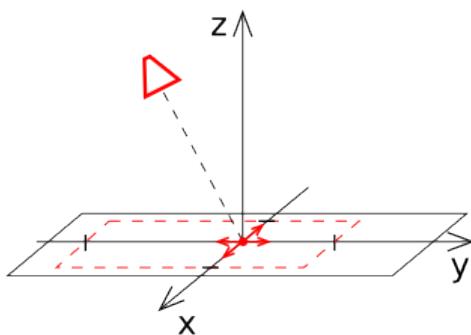


Hover Type Camera



Camera Move Style > Camera Limits > Horizontal Translation Limits Limit movement of the pivot point along the Blender's world X axis. Allowable values: $\text{Min} \leq \text{Max}$. Default values: $\text{MinX} = -10$, $\text{MaxX} = 10$. Disabled by default.

Camera Move Style > Camera Limits > Vertical Translation Limits Limit movement of the pivot point along the Blender's world X axis. Allowable values: $\text{Min} \leq \text{Max}$. Default values: $\text{MinY} = -10$, $\text{MaxY} = 10$. Disabled by default.



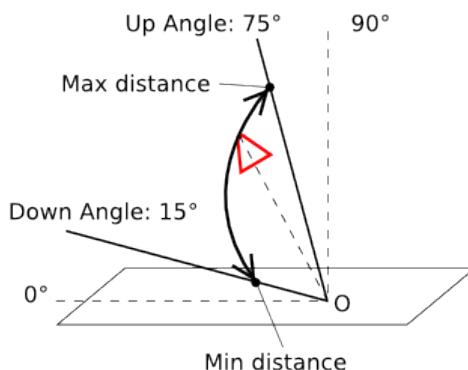
Camera Move Style > Camera Limits > Use Zooming Allows user to zoom the camera in and out from the pivot point. If this parameter is disabled, the camera will always remain at a fixed distance from the pivot point. Disable by default.

Camera Move Style > Camera Limits > Distance Limits Set minimum and maximum distances from the camera to the point of intersection between the camera's view direction and the horizontal reference plane (Blender's world XOY plane by default). Allowable values: $\text{Min} \leq \text{Max}$. Default values: Min = 1, Max = 10. Disabled by default.

Camera Move Style > Camera Limits > Vertical Rotation Limits Limits the camera's angle of elevation (an angle between the camera's sight line and a horizontal plane. Possible value interval: $0^\circ \leq \text{Down} \leq \text{Up} \leq 90^\circ$. Default values: Down = 0° , Up = 90° .

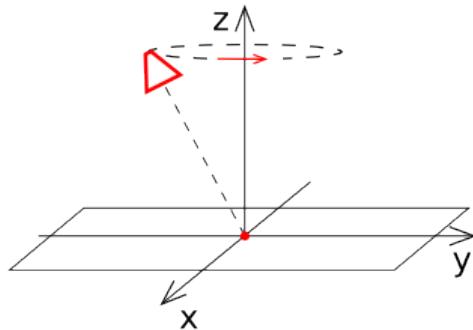
Camera Move Style > Camera Limits > Zero Level A distance between reference plane and the coordinates origin. Set to zero by default.

If the Blend4Web > Use Zooming setting is turned on, the limits for distance and inclination angle will be applied simultaneously, to define the camera movement path in the vertical plane.

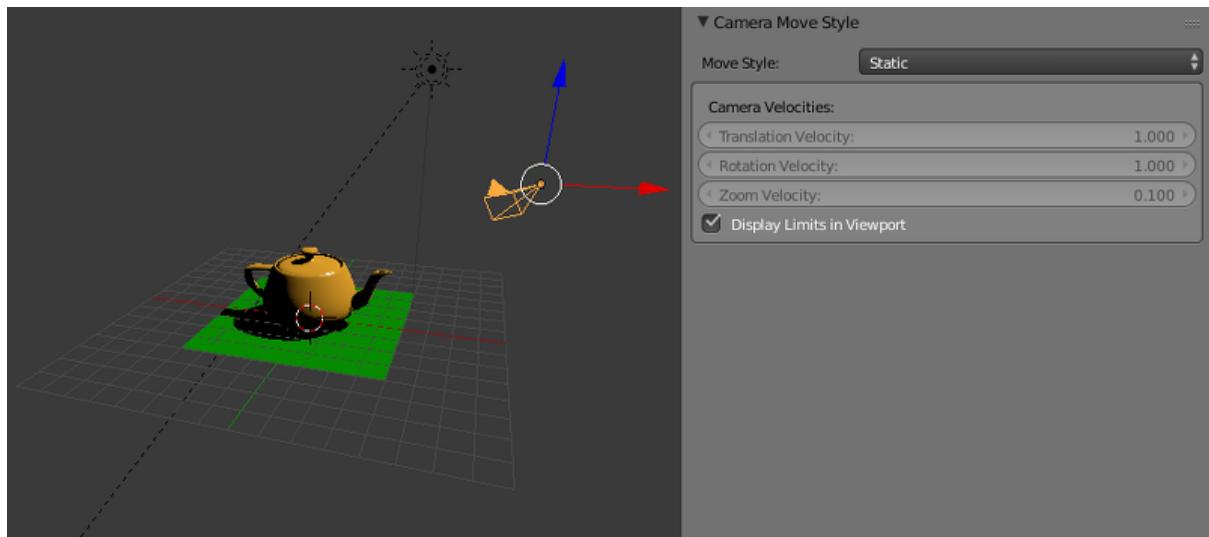


Setting incorrect distance or elevation angle limits will disable this parameter.

Camera Move Style > Camera Limits > Use Horizontal Rotation Allow camera rotation in Blender's XOY plane around to the pivot point. Enabled by default.



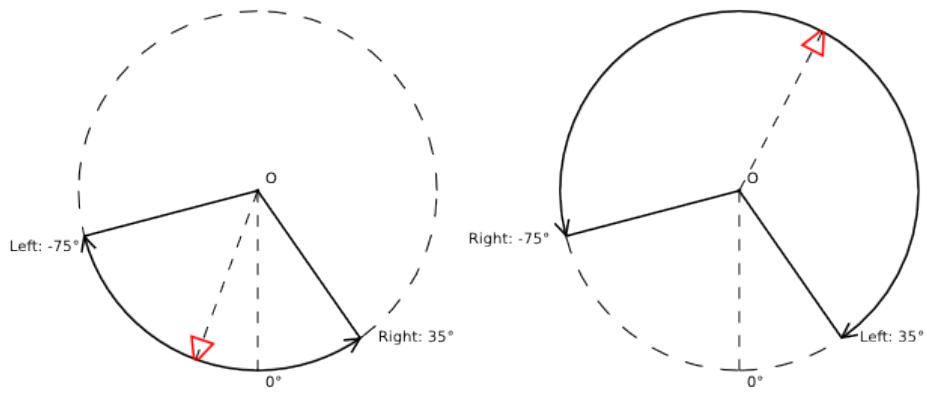
Static Camera



This type of camera does not have limits, as it can't be directly controlled by the user.

Peculiarities of Limiting Settings

- For EYE/TARGET cameras, interchanging Left/Right or Down/Up values results in movement along the opposite arc of a circle.



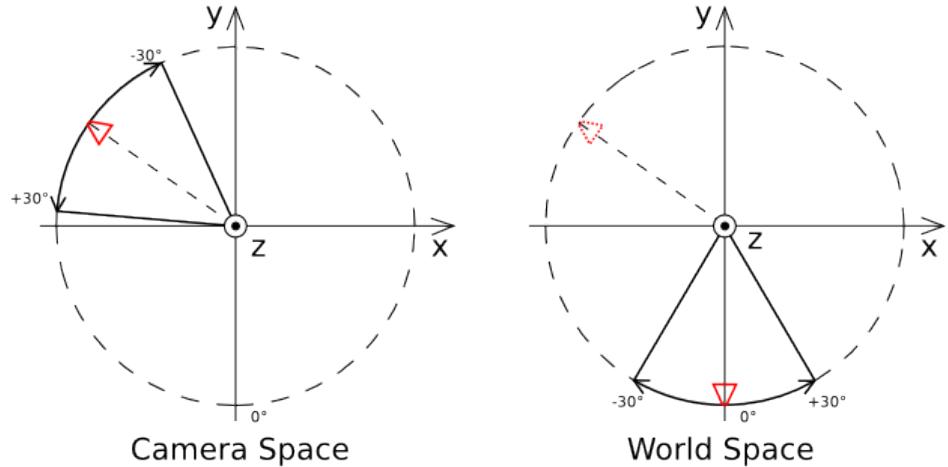
- When limiting the camera's horizontal and vertical rotation, it is possible to choose between the following coordinate spaces:

Camera Space All angles are measured relative to the initial camera position and orientation.

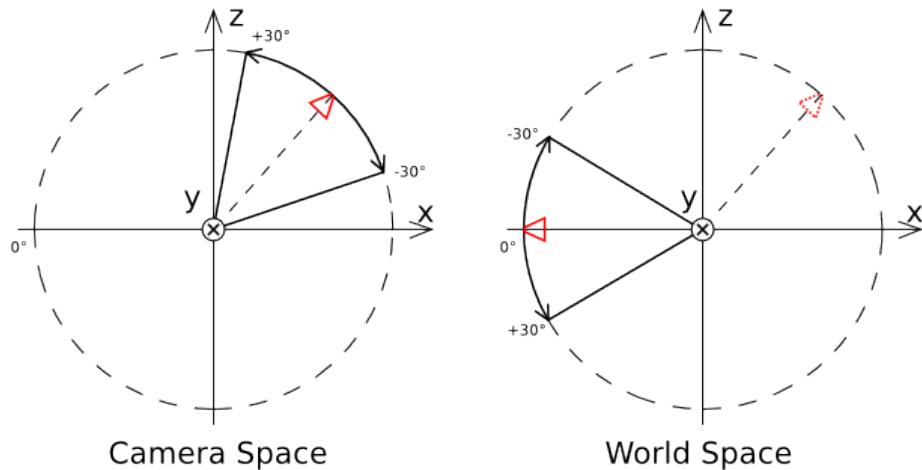
World Space Horizontal angles are measured relative to the Y axis in world space; vertical angles are measured relative to the Blender's XOY plane in world space.

Default value: Camera Space.

Horizontal limits by the example of the TARGET camera:

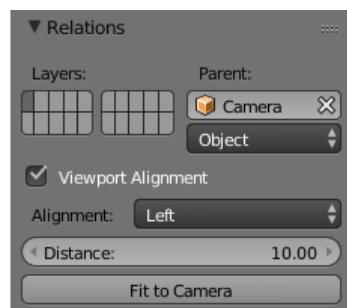


Vertical limits by the example of the TARGET camera:

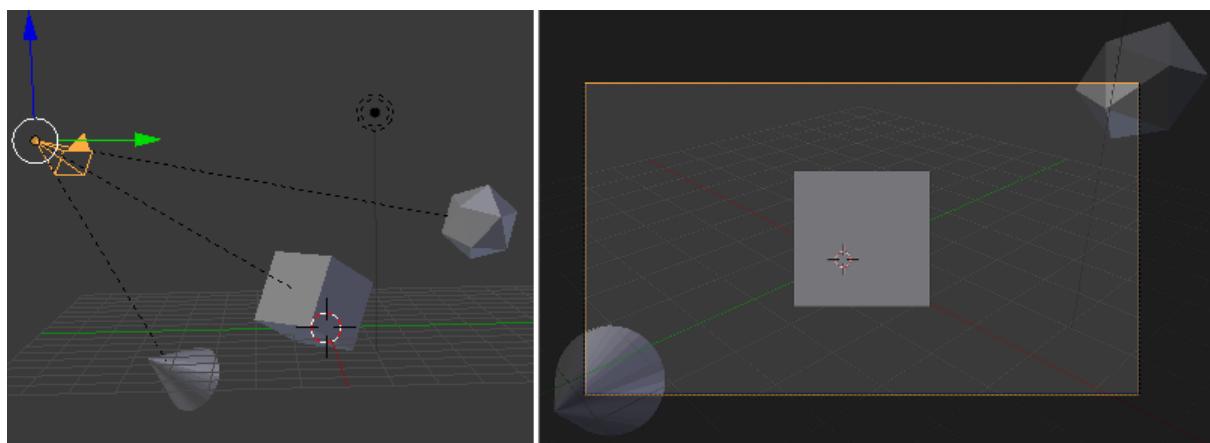


Viewport Alignment

If an object is parented to a camera, the set of parameters known as Viewport Alignment are available under the Relations tab on the Object panel.



Using these settings, the object can be aligned to the active camera, so that the object will remain in the same place of the screen regardless of the position and rotation of the camera. This is especially useful for creating UI elements.



This picture shows an example of how the Viewport Alignment option can be used.

Viewport Alignment This parameter enables and disables all the following parameters.
It is disabled by default.

Alignment This parameter specifies what side of the screen the object will be aligned with. It can have one of the following values.

- Top-Left - aligns the object in the top left corner of the viewport
- Top - aligns the object in the top of the viewport
- Top-Right - aligns the object in the top right corner of the viewport
- Left - aligns the object in the left side of the viewport
- Center - aligns the object in the center of the viewport
- Right - aligns the object in the right side of the viewport
- Bottom-Left - aligns the object in the bottom left corner of the viewport
- Bottom - aligns the object in the bottom of the viewport
- Bottom-Right - aligns the object in the bottom right corner of the viewport.

Default value is Center.

Distance This specifies the distance between the aligned object and the camera. It is set to 10 by default.

Fit to Camera By pressing this button, you can make the aligned object look the same way in the Blender viewport as it will look in the engine. It can be used to preview alignment results.

Camera Controls API

Detailed API module documentation: [link](#).

A camera's position and its spatial orientation are defined by the control mode. In the EYE, TARGET and HOVER modes, control mode has several limitations: for example, the camera's vertical axis is always aligned along the world's Y-axis and the camera is constantly focused on the pivot point. The STATIC mode camera has no such limitations, which makes it more suitable for cases when you need more complete control, such as procedural animation.

Main functions for camera control are located in the `camera` module. Some of them (intended for a specific control mode) have names which start with an appropriate prefix: `static_`, `eye_`, `target_` and `hover_`. Other functions can be used in any mode.

Control Mode Setup

To change the control mode and to completely define a camera's behavior, the following methods can be used: `static_setup()`, `eye_setup()`, `target_setup()` and `hover_setup()`.

These methods receive an object that contains a set of optional parameters, which can be used to set the camera's position, rotation, available limits and so on.

```

var camera = m_scenes.get_active_camera();
var POS = new Float32Array([1,1,1]);
var LOOK_AT = new Float32Array([0,0,0]);
var EYE_HORIZ_LIMITS = { left: Math.PI/4, right: -Math.PI/4 };
var EYE_VERT_LIMITS = { down: -Math.PI/4, up: Math.PI/4 };
var TARGET_DIST_LIMITS = { min: 1, max: 10 };
var HOVER_DIST_LIMITS = { min: 1, max: 10 };
var HOVER_ANGLE_LIMITS = { down: 0, up: -Math.PI/4 };
var HOVER_HORIZ_TRANS_LIMITS = { min: -5, max: 3 };
var HOVER_VERT_TRANS_LIMITS = { min: -1, max: 1 };

...

// setup STATIC camera by defining the new position and the new look-at point
m_cam.static_setup(camera, { pos: POS, look_at: LOOK_AT });

// setup STATIC camera by defining the new look-at point and keeping the existing position
m_cam.static_setup(camera, { look_at: LOOK_AT });

// setup STATIC camera by defining the new position and keeping the existing orientation
m_cam.static_setup(camera, { pos: POS });

// setup EYE camera with horizontal rotation limits
m_cam.eye_setup(camera, { pos: POS, look_at: LOOK_AT,
    horiz_rot_lim: EYE_HORIZ_LIMITS });

// setup EYE camera with vertical rotation limits
m_cam.eye_setup(camera, { pos: POS, look_at: LOOK_AT,
    vert_rot_lim: EYE_VERT_LIMITS });

// setup TARGET camera with distance limits and panning mode
m_cam.target_setup(camera, { pos: POS, pivot: LOOK_AT,
    dist_lim: TARGET_DIST_LIMITS, use_panning: true });

// setup HOVER camera on a fixed distance (without zooming) relatively to its pivot
m_cam.hover_setup(camera, { pos: POS, pivot: LOOK_AT });

// setup HOVER camera with zooming (distance + angle limits)
m_cam.hover_setup(camera, { pos: POS, pivot: LOOK_AT,
    dist_lim: HOVER_DIST_LIMITS, hover_angle_lim: HOVER_ANGLE_LIMITS });

// setup HOVER camera with translation limits
m_cam.hover_setup(camera, { pos: POS, pivot: LOOK_AT,
    horiz_trans_lim: HOVER_HORIZ_TRANS_LIMITS,
    vert_trans_lim: HOVER_VERT_TRANS_LIMITS });

// setup HOVER camera with horizontal rotation enabled
m_cam.hover_setup(camera, { pos: POS, pivot: LOOK_AT, enable_horiz_rot: true });

```

The main characteristic of the HOVER type camera is the fact that the distance and elevation angle limits define a certain path of movement toward the pivot point. The camera will always be located on this path, so its final position can be different from the one returned by the `hover_setup()` method (basically, it is projected onto the path). If you need to set the camera's position on some path, you can use the `hover_setup_rel()` method to set the relative limits.

```
var camera = m_scenes.get_active_camera();
var POS = new Float32Array([1,1,1]);
var PIVOT = new Float32Array([0,0,0]);

...
// setup HOVER camera with maintaining the given camera position and
// ability to zoom-in and zoom-out equally
m_cam.hover_setup_rel(camera, { pos: POS, pivot: PIVOT, dist_interval: 2,
    angle_interval: Math.PI/4 });
```

The current camera mode can be checked with the `is_static_camera()`, `is_eye_camera()`, `is_target_camera()`, `is_hover_camera()` or `get_move_style()` methods:

```
var camera = m_scenes.get_active_camera();

...
if (m_cam.is_static_camera(camera))
    console.log("STATIC camera!");

if (m_cam.get_move_style(camera) == m_cam.MS_EYE_CONTROLS)
    console.log("EYE camera!");
```

Note: If you need to change only some aspects of the camera's behavior, you can use the methods described below.

Translation and Rotation

To rotate a STATIC type camera, you need to use the `static_get_rotation()` and `static_set_rotation()` methods. In these methods, rotation is defined by a quaternion.

```
var camera = m_scenes.get_active_camera();

...
// rotation through a quaternion
var _quat_tmp = new Float32Array(4);
var old_quat = m_cam.static_get_rotation(camera, _quat_tmp);
var new_quat = m_quat.rotateX(old_quat, Math.PI/2, old_quat)
m_cam.static_set_rotation(camera, new_quat);
```

For the EYE, TARGET and HOVER type cameras, rotation is performed in spherical coordinates by using the `eye_rotate()`, `target_rotate()`, `hover_rotate()` and `rotate_camera()` methods:

```
var camera = m_scenes.get_active_camera();

...
// rotate by given delta angles
m_cam.eye_rotate(camera, Math.PI/6, Math.PI/2);

// set absolute rotation in world space
m_cam.eye_rotate(camera, Math.PI/6, Math.PI/2, true, true);
```

Current spherical coordinates of the camera can be obtained by using the `get_camera_angles()` method:

```
var camera = m_scenes.get_active_camera();
var _vec2_tmp = new Float32Array(2);

...
// get camera orientation in spherical coordinates
var angles = m_cam.get_camera_angles(camera, _vec2_tmp);
phi = angles[0];
theta = angles[1];
```

Access to the position of the camera can be obtained by using the `get_translation()` and `set_translation()` methods. For the TARGET and HOVER modes this means simultaneous translation of an entire model, including camera position and pivot point.

```
var camera = m_scenes.get_active_camera();
var _vec3_tmp = new Float32Array(3);

...
// get camera position
var pos = m_cam.get_translation(camera, _vec3_tmp);

// set new position
var new_pos = m_vec3.set(1, 0, 2, pos);
m_cam.set_translation(camera, new_pos);
```

Additional camera translation methods:

- `target_set_trans_pivot()`, `target_set_pivot_translation()`, `hover_set_pivot_translation()` - sets the position and the pivot point of the camera at the same time;
- `target_get_distance()`, `target_set_distance()` - translation based on the distance to the pivot point;
- `static_set_look_at()`, `eye_set_look_at()` - sets the camera's position and point of

sight simultaneously;

Note: Because a camera is a scene object, `transform` module methods can also be used. But, as every control mode influences camera movement differently, results may differ from those expected.

Limit Setup

Camera limits are available for the EYE, TARGET and HOVER type cameras. To set a specific limit, you need to use a specific method:

EYE	TARGET	HOVER
<code>eye_get_horizontal_limits()</code> <code>eye_set_horizontal_limits()</code> <code>eye_get_vertical_limits()</code> <code>eye_set_vertical_limits()</code> <code>get_set_distance_limits()</code> <code>tar_get_get_horizontal_limits()</code> <code>tar_get_set_horizontal_limits()</code> <code>target_get_vertical_limits()</code> <code>target_set_vertical_limits()</code> <code>target_get_pivot_limits()</code> <code>target_set_pivot_limits()</code>		<code>hover_get_distance_limits()</code> <code>hover_set_distance_limits()</code> <code>hover_get_vertical_limits()</code> <code>hover_set_vertical_limits()</code> <code>hover_get_horiz_trans_limits()</code> <code>hover_set_horiz_trans_limits()</code> <code>hover_get_vert_trans_limits()</code> <code>hover_set_vert_trans_limits()</code>

```

var camera = m_scenes.get_active_camera();
var _limits_tmp = {};
var EYE_HORIZ_LIMITS = { left: Math.PI/4, right: -Math.PI/4 };

...
// get limits
m_cam.eye_get_horizontal_limits(camera, _limits_tmp);

// set limits
m_cam.eye_set_horizontal_limits(camera, EYE_HORIZ_LIMITS);

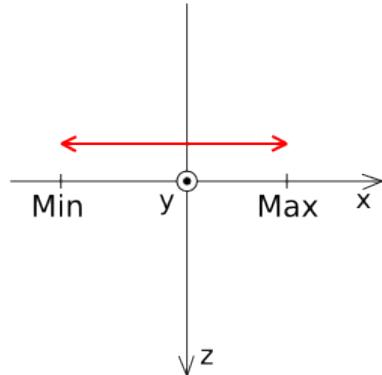
```

Limit presence can be checked by using the `has_distance_limits()`, `has_horizontal_rot_limits()`, `has_vertical_rot_limits()`, `has_horizontal_trans_limits()` and `has_vertical_trans_limits()` methods.

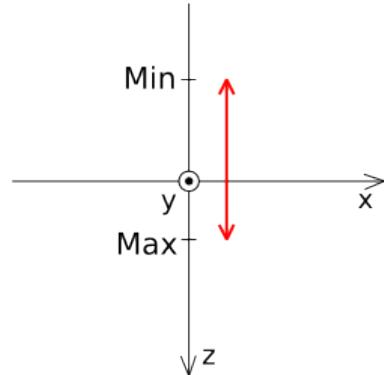
Note: In HOVER mode, the camera always has distance and elevation angle limits. In case these limits were not set, they are automatically calculated to lock the camera in the current position relative to the pivot point.

Note: When setting translation limits for the HOVER camera, the values for the limits will correspond to the engine's coordinate axis as follows:

- Horizontal Translation Limits - X axis
- Vertical Translation Limits - Z axis



Horizontal limits



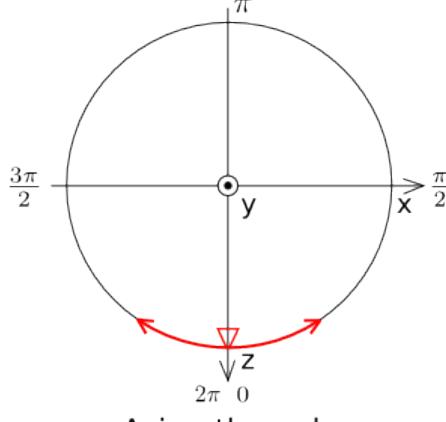
Vertical limits

Note: Enabled camera limits can influence its position and spatial orientation set by the API functions.

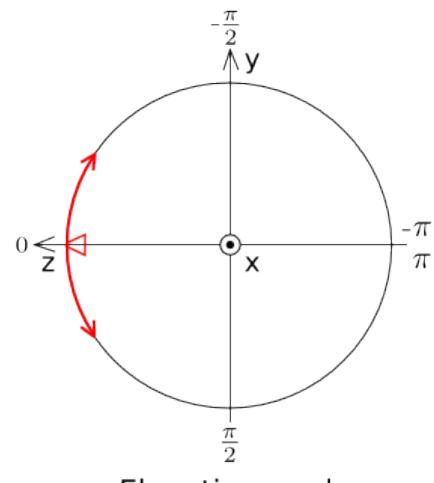
Angular Coordinates Count

When working with the camera's APIs (rotation, setting limits), all angles are defined in the engine's coordinate system as follows:

For the TARGET/HOVER camera:

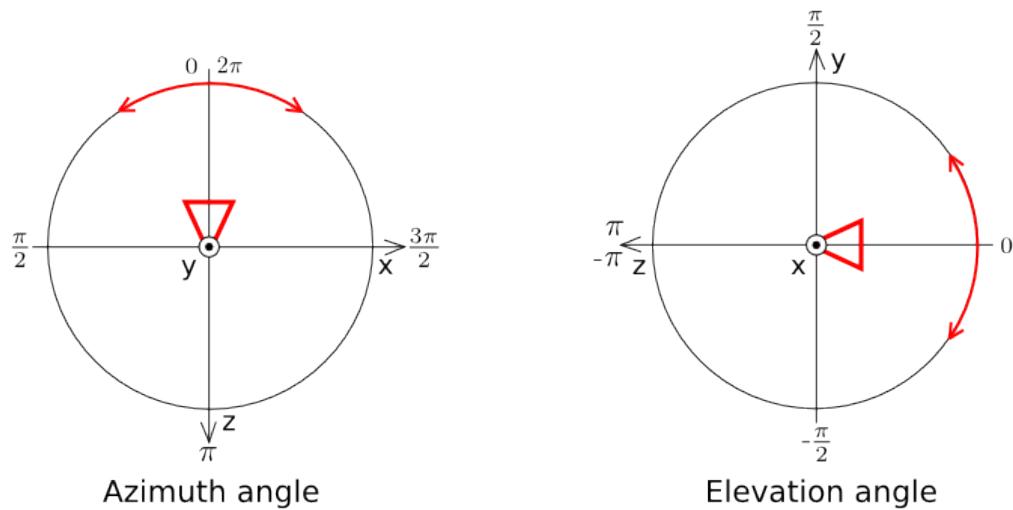


Azimuth angle



Elevation angle

For the EYE camera:



Materials

Table of Content

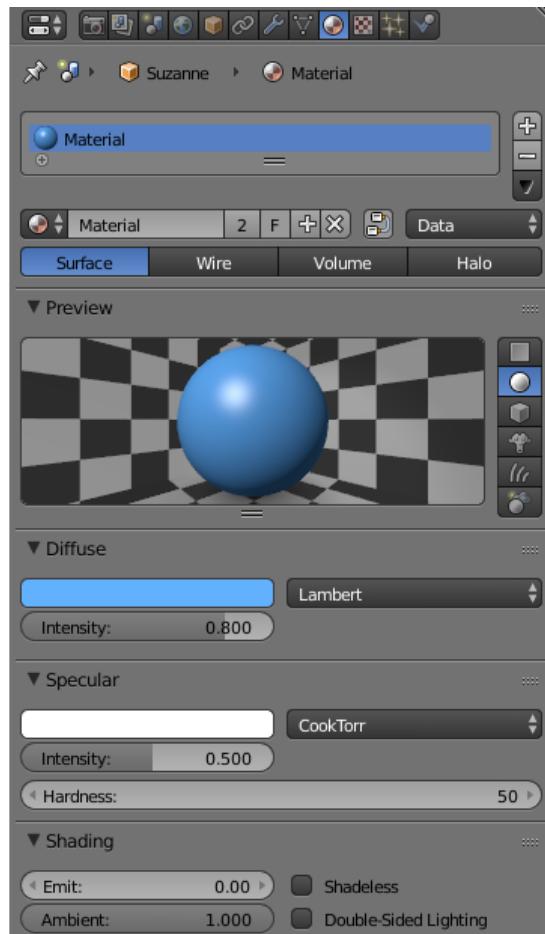
- Materials
 - Lighting Parameters
 - Transparency
 - * Types
 - * Additional Settings
 - Reflection
 - * Static Reflection
 - * Dynamic Reflection
 - * Fresnel effect for reflection
 - Rendering Properties
 - Viewport Properties
 - Engine Specific Parameters
 - Halo Materials
 - * Activation
 - * Additional Settings

Materials describe the object surface's response to light and also contain information about its transparency, reflectivity, physical parameters and so on.

Meshes can have one or more materials. In case of multiple materials they can be assigned to different polygons in the Edit Mode. To do this select the needed polygons, select the needed material from the list and click the Assign button.

The following material types are supported: Surface, Halo.

Lighting Parameters



Diffuse > Color Diffuse light color. The default value is (0.8, 0.8, 0.8). It may interact with the diffuse map color.

Diffuse > Intensity Diffuse light intensity. The default value is 0.8.

Diffuse > Shader Diffuse shading algorithm. The default value is Lambert.

Specular > Color Specular light color. The default value is (1.0, 1.0, 1.0). It may interact with the specular map color.

Specular > Intensity Specular light intensity. The default value is 0.5.

Specular > Hardness Exponent in the specular shading calculation formula. The default value is 50. Note that the formula used in the engine differs slightly from the Blender's one.

Specular > Shader Specular shading algorithm. The default value is CookTorr.

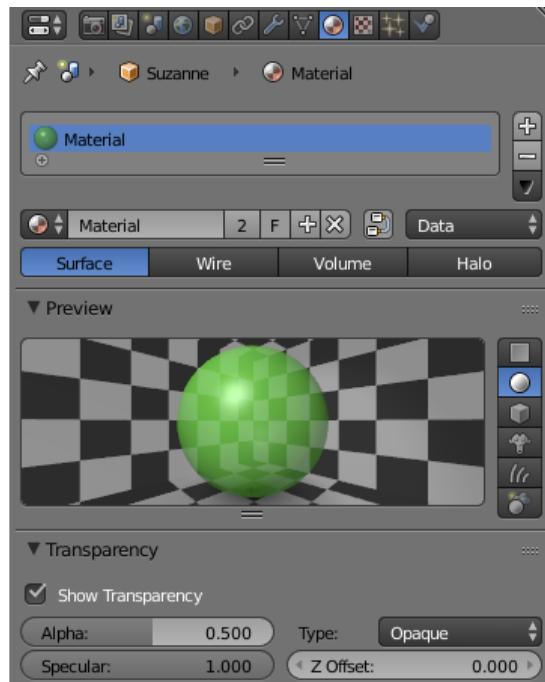
Shading > Emit Emission intensity. The default value is 0.0.

Shading > Ambient Ambient influence factor on material. The default value is 1.0.

Shading > Shadeless When enabled, a material doesn't react to light. Disabled by default.

Shading > Double-Sided Lighting Enables the double-sided lighting mode. This option is useful for non-transparent objects with a single-layered mesh.

Transparency



Types

Transparency implementation type can be selected in the Transparency menu on the Properties > Material panel.

The engine supports the following transparency implementation types (sorted in the ascending order by performance):

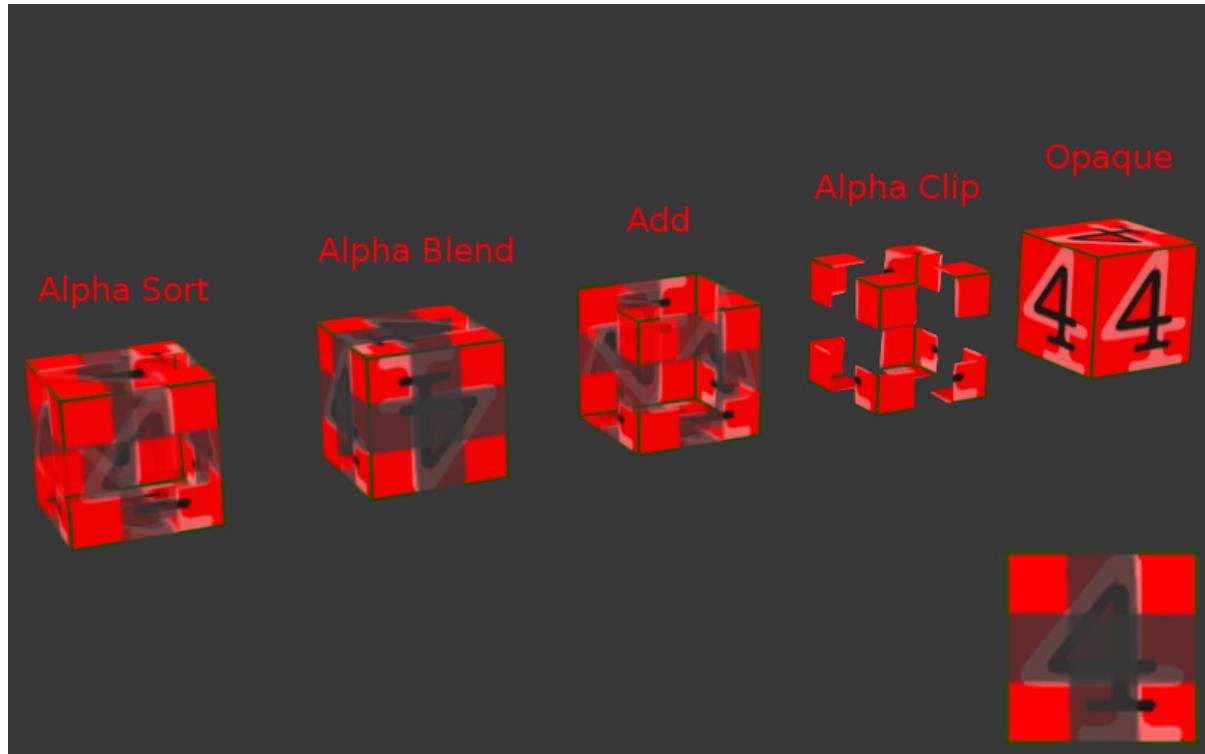
Alpha Sort Transparent with a gradient. The engine sorts the triangles by camera distance in order to render overlapping transparent surfaces correctly. This operation is computationally expensive. It is recommended to use this feature for closed transparent geometry (bottle, car glass etc).

Alpha Blend Transparent with a gradient. The sorting of triangles is not performed. It is recommended to use this feature for unclosed transparent geometry (water surface, decals).

Add Transparent with a gradient. The sorting of triangles is not performed. The engine disables writing to the depth buffer which causes transparent surfaces to be rendered in arbitrary order. It is recommended to use this feature for effects (particle systems, glowing beams).

Alpha Clip Transparent without a gradient. The engine discards pixels if their alpha is less than 0.5. The sorting of triangles is not performed. It is recommended to use

this feature with a mask texture to visualize smaller details (tree leaves, grass).
Opaque Non-transparent. Alpha is ignored. This is the default value.



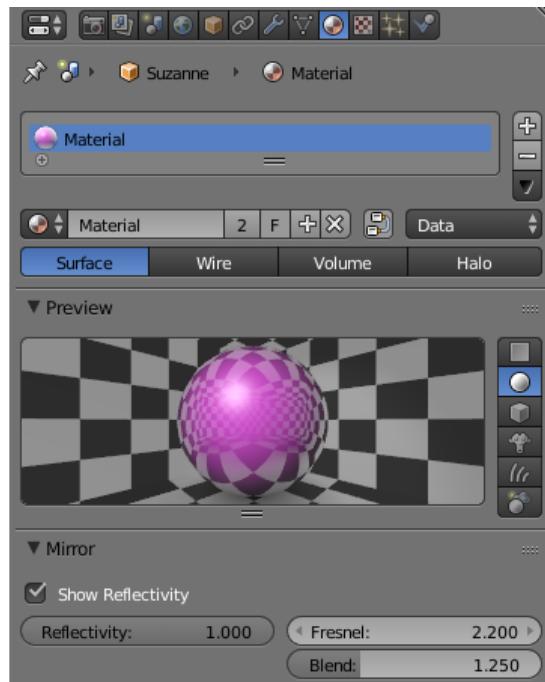
Additional Settings

Transparency > Show Transparency Enabling the transparency checkbox is required for viewing transparent objects in Blender. The engine ignores this option - the Alpha Blend option is used instead.

Transparency > Alpha Material transparency level. The engine ignores this parameter (in contrast to Blender) if there is a diffuse texture - the alpha channel values of a texture are used instead.

Transparency > Z Offset This option explicitly specifies relative positioning order of objects with different materials with the purpose of depth sorting. The option can take both negative and positive values. The more distant the object is the lesser parameter value should be to provide correct rendering. The default value is 0.0.

Reflection



Static Reflection

A surface reflects the same image no matter how the environment changes. For activation simply use the [mirror map](#).

See also:

[Fresnel effect for reflection](#)

Dynamic Reflection

A surface reflects the selected objects in their current position. The engine supports planar and spherical reflections.

Activation

1. Check **Reflections** setting on the **Render > Reflections and Refractions** panel.
2. For reflective objects enable the **Reflective** option on the **Object > Reflections** panel.
 - For planar reflections, set the **Object > Reflections > Type** property to **Plane**. After that, add an empty object to be used as a reflection plane by executing for example **Add > Empty > Single Arrow**. Rename it for convenience and specify its name in the **Reflection plane** field of the reflective object.

- For cube-mapped reflections, set the Object > Reflections > Type property to Cube.
3. For the needed materials of the reflective objects, set the Material > Mirror > Reflectivity value.
 - Mirror > Reflectivity > Show Reflectivity is required for displaying reflections on objects in Blender. The engine ignores this option.
 4. For the reflexible objects, enable the Reflexible checkbox on the Object > Reflections panel.

Note: It is also recommended to enable the World > Environment Lighting checkbox.

Limitations

Normal maps and shadows are ignored in the reflected image for optimization purposes.

See also:

[Fresnel effect for reflection](#)

Fresnel effect for reflection

The Fresnel effect manifests itself as the dependency of the intensity of passing and reflected light on the incidence angle. If the angle of incidence is close to zero (i.e. light falls almost at right angle to the surface) the passing light portion is large and the reflected light portion is small. On the contrary if the angle of incidence is close to 90 degrees (i.e. light falls almost parallel to the surface) almost all light is reflected.

The engine uses the approximate Schlick's formula:

$$R = R_0 + (1 - R_0)(1 - \cos \theta)^N, \text{ where}$$

R - reflection coefficient,

R_0 - reflection coefficient in case of viewing at a right angle to the surface (i.e. when $\theta = 0$),

θ - angle of incidence (which is equal to the angle of reflection under which light enters the camera), it is calculated by the engine in real-time,

N - exponent.

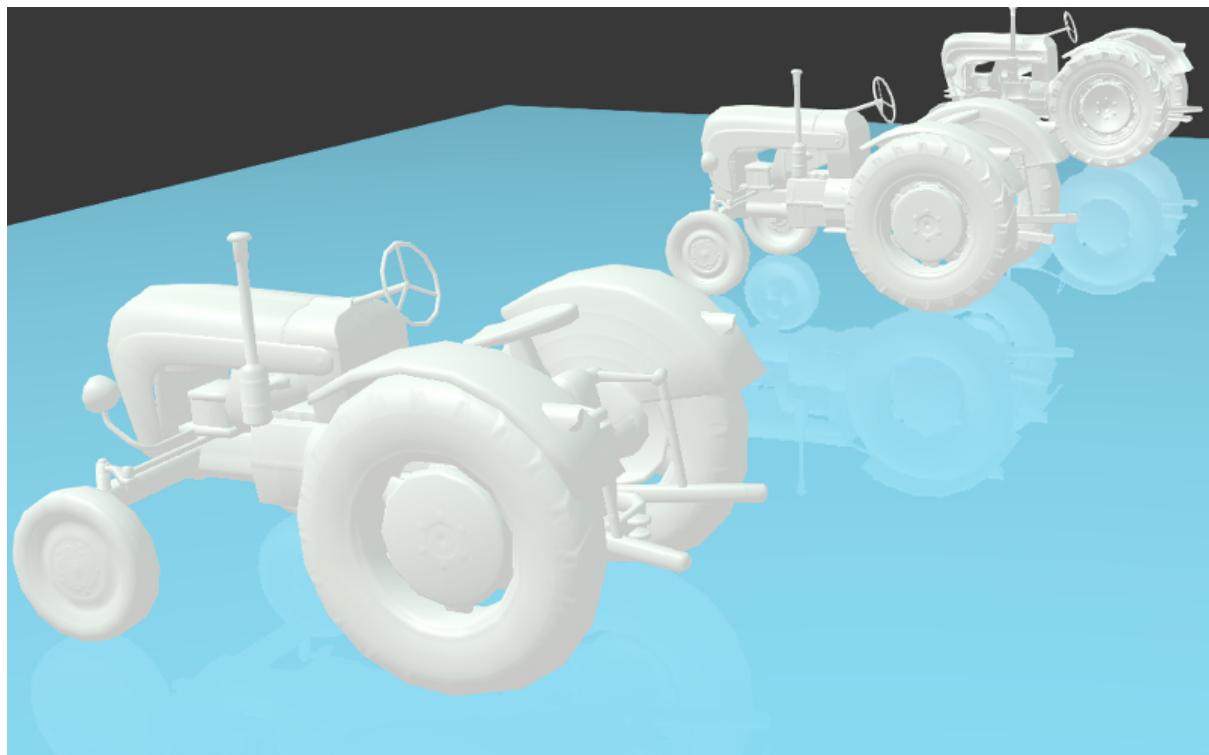
Settings

Fresnel effect can be set up both for static and dynamic reflection.

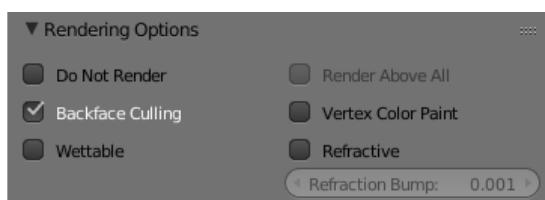
Material > Mirror > Fresnel Fresnel power for reflection. This is the N exponent in the Schlick's formula. In Blender it is limited to values from 0 to 5. If this parameter is equal

to zero the Fresnel effect is not observed and the full reflection at all angles occurs. If this parameter is greater than zero, the material is less reflective when viewing surfaces at angles which are close to the right angle. The bigger this parameter is the bigger is the angle deviation from the right angle for which the Fresnel effect is observed.

Material > Mirror > Blend Fresnel factor for reflection. It is reduced to R_0 in the Schlick's formula by the following expression: $R_0 = 1 - \text{Blend} / 5$. In Blender it is limited to values from 0 to 5. This parameter defines the Fresnel effect intensity: the bigger the Blend factor is, the more is the Fresnel effect influence. If it is equal to zero the Fresnel effect is not observed.



Rendering Properties



Material > Rendering Options > Do not Render Disable rendering of this object.

Material > Rendering Options > Backface Culling When enabled, polygons' back faces are not rendered by the engine. Enabled by default.

Material > Rendering Options > Wettable Water wetting effect is activated for the material.

See also:

[Water](#)

Material > Rendering Options > Render Above All Material is rendered on top of all scene objects. Transparency type with a gradient is required (Add, Alpha Blend“ or Alpha Sort).

Material > Rendering Options > Vertex Color Paint Mesh vertex color is used instead of the material diffuse color when the checkbox is enabled.

Material > Rendering Options > Refractive Make object refractive. Perturbation factor can be set with the option Refraction Bump on the Refraction Settings panel. The default value is 0.001.

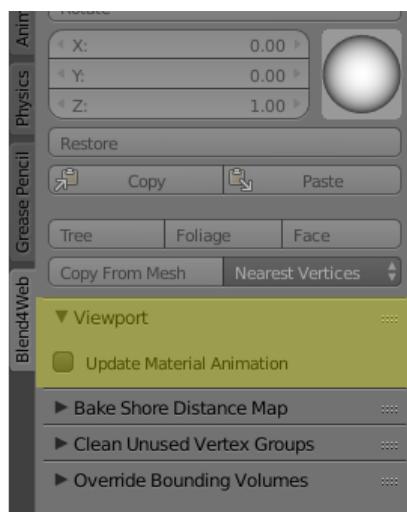
Note: In order to use this effect, select ON or AUTO on the Render > Reflections and Refractions > Refractions panel. The object must have Alpha Blend transparency type.

See also:

[Transparency](#)

Viewport Properties

The Viewport section can be found on the Blend4Web panel.



[Update Material Animation](#)

Update animated shader in Blender Viewport.

Engine Specific Parameters



Material > Water Special material for [water](#) rendering.

Material > Terrain Dynamic Grass Material is used for [grass](#) rendering.

Material > Collision A special material for collision geometry.

See also:

[Physics](#)

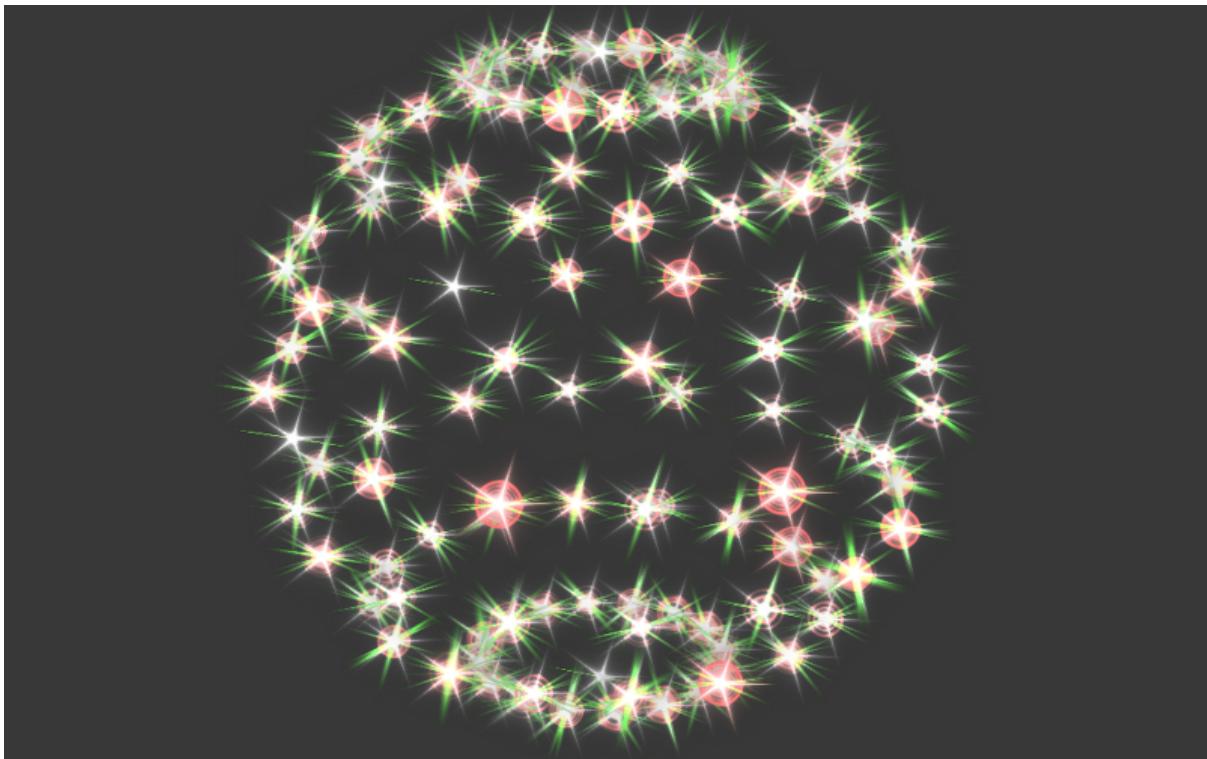
Material > Export Options > Do Not Export Material is not to be exported.

Halo Materials

Halo materials are used in particle systems and in static meshes. Using the halo in static meshes is described below.

Activation

Select the Halo type under the Materials tab. It's also recommended to select the transparency type with a gradient (Add, Alpha Blend or Alpha Sort).



Additional Settings

Halo > Alpha Material transparency factor. The default value is 1.0 (non-transparent).

Halo > Color Material color. The default value is (0.8, 0.8, 0.8) (almost white).

Halo > Size Particle size. The default value is 0.5.

Halo > Hardness Exponent for computing the gradient. Affects visible dimensions of particles. The default value is 50.

Halo > Rings Use rings. Relative quantity and color can be set up.

Halo > Lines Use lines. Relative quantity and color can be set up.

Halo > Star Tips Use stars. The quantity of edges can be set up.

Halo > Special: Stars Enables the starry sky rendering mode. The mesh is fixed relative to the camera. For the Sun lamp it is also required to enable the Lamp > Dynamic Intensity checkbox. Applications should set up the hours of darkness via API.

Halo > Blending Height Height range for the fading of stars.

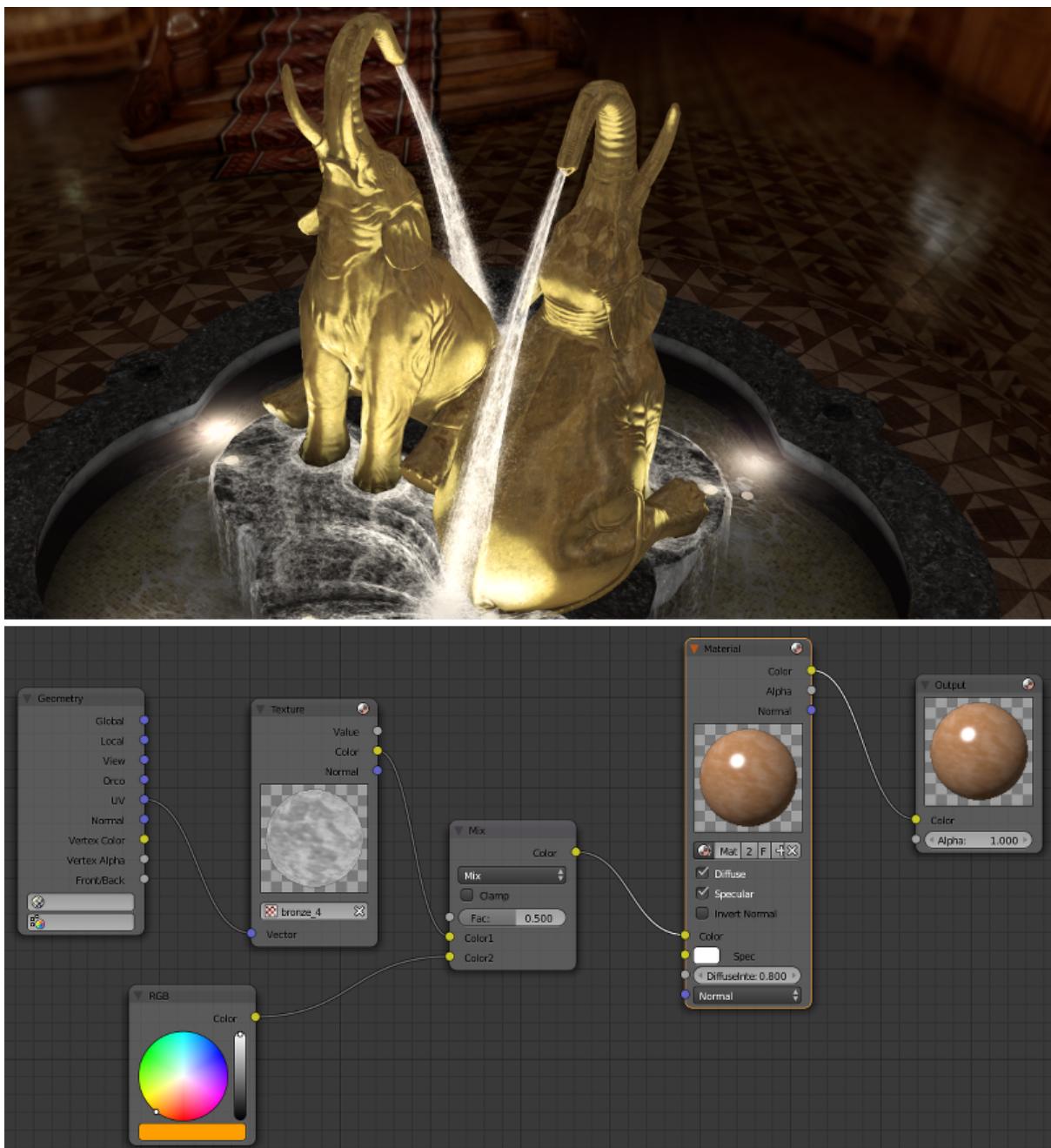
Halo > Minimum Height Minimum height in the object's local space at which stars are visible.

Node Materials

Table of Content

- Node Materials
 - Standard Nodes
 - Engine Specific Nodes
 - * Replace (B4W_REPLACE)
 - * Clamp (B4W_CLAMP)
 - * Time (B4W_TIME)
 - * Vector View (B4W_VECTOR_VIEW)
 - * Parallax (B4W_PARALLAX)
 - * Translucency (B4W_TRANSLUCENCY)
 - * Refraction (B4W_REFRACTION)
 - * Levels of Quality (B4W_LEVELS_OF_QUALITY)
 - * Smoothstep (B4W_SMOOTHSTEP)
 - * Glow Output (B4W_GLOW_OUTPUT)
 - * Reflect (B4W_REFLECT)
 - * Linear to SRGB and SRGB to Linear (Deprecated)

Shader nodes extend significantly the potential of Blender's standard materials by means of presenting shading as a batch of basic transformations.



Standard Nodes

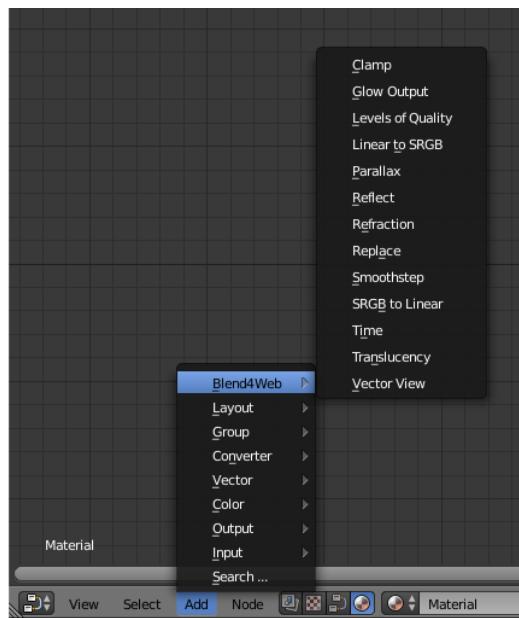
Blend4Web supports all standard Blender nodes, but some of them do not work fast enough and are not recommended to use in real time applications. Creating very complex materials, especially using large numbers of Geometry and Texture nodes, is also not recommended.

Cycles nodes are only partially supported: in most cases they will not work in Blend4Web the same way they do in Blender. They also might not work at all or even cause material in which they are used to work incorrectly. However, using Cycles nodes will not cause instabilities in the application workflow.

Standard node performance and degree of support is described in the [table](#).

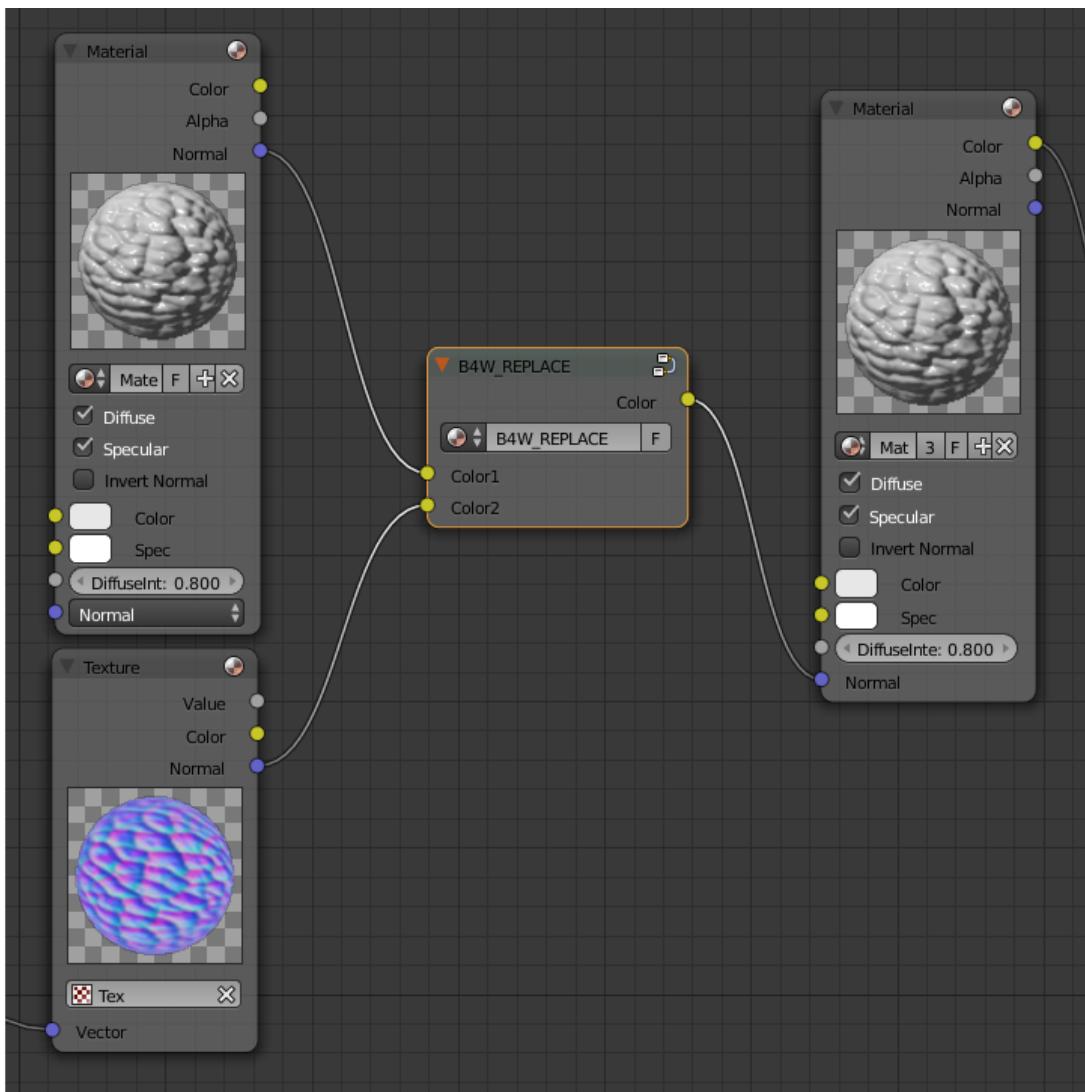
Engine Specific Nodes

Engine-specific nodes extend functionality of the standard nodes to support extra features. These nodes are created as node groups (Node Groups or Node Tree) with specially determined names and input formats. For convenience, all special nodes are added to a blend file when it is opened.



Replace (B4W_REPLACE)

The node replaces the inputs depending on the working environment (i.e. Blender viewport or Blend4Web). When working in Blender the Color1 input is connected to the Color output and the Color2 input is ignored. On the contrary when working in the engine the inputs are interchanged (the Color1 one is ignored and the Color2 one is connected to the output). The node is intended to display one node structure in the viewport and another - in the engine.



As a rule it is used for normal mapping. Blender's node materials do not support a tangent space of coordinates. Therefore, the only possible method to display normal maps in the viewport correctly is their usage inside the Material nodes.

Input Parameters

Color1 Node setup that will be visible in the Blender viewport.

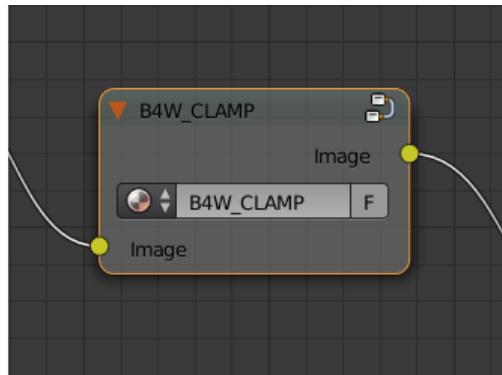
Color2 Node setup that will be visible in the Blend4Web engine.

Output Parameters

Color Should be connected to the Color socket of the Material or Extended Material node.

Clamp (B4W_CLAMP)

The node limits the output value. As a result all the output vector components take values from 0 to 1 inclusive.



Input Parameters

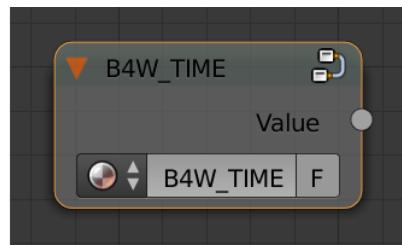
Image Incident vector.

Output Parameters

Image Clamped vector.

Time (B4W_TIME)

Provides the timeline counting from the engine start (in seconds). Can be used for animating any parameters in node materials, such as UV coordinates, mixing factors, transparency etc.



Input Parameters

None.

Output Parameters

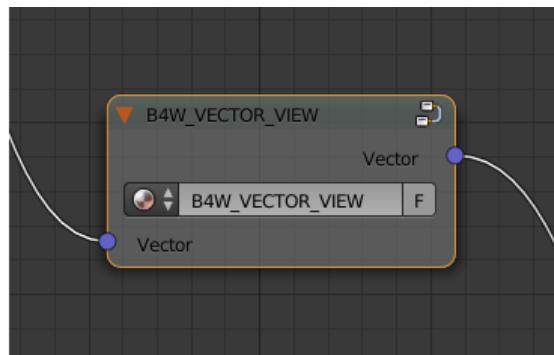
Value Time (in seconds) elapsed from the engine startup.

See also:

[Animation of Value and RGB Nodes](#)

Vector View (B4W_VECTOR_VIEW)

The node transforms a vector into the camera's space of coordinates. Transformation is necessary because the engine defines most vectors in the world space of coordinates. If normal vector is being transformed by this node it should be used only for effects and not for connecting to the output of the Material or Extended Material nodes.



Input Parameters

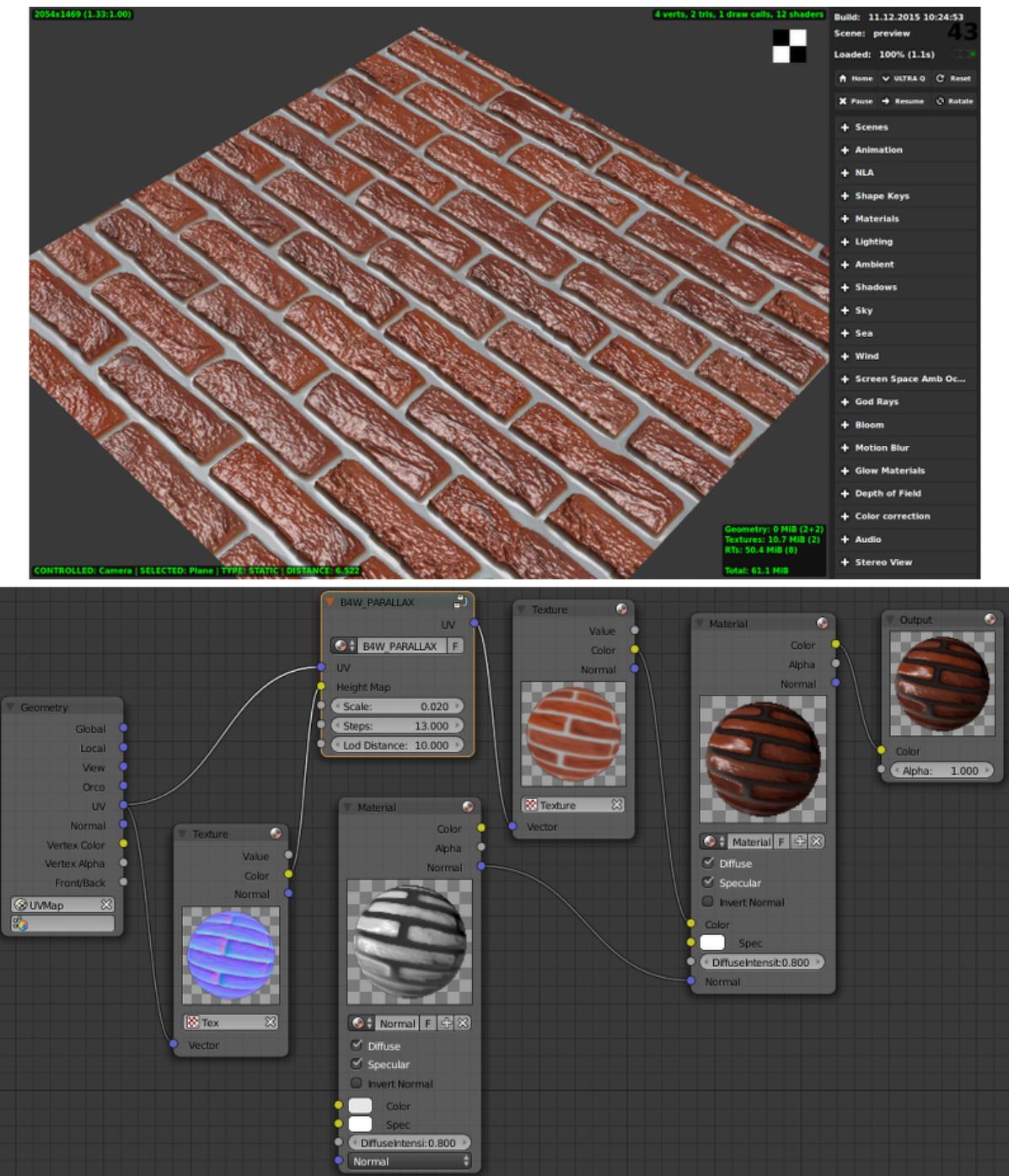
Vector Vector coordinates in the world-space.

Output Parameters

Vector Vector coordinates in the camera-space.

Parallax (B4W_PARALLAX)

The node implements the texture coordinates offset using a height map.



Input Parameters

UV Source texture coordinates.

Height Map RGBA texture with a height map packed into the alpha channel.

Scale Texture coordinates offset factor.

Steps The number of steps for iterative generation of texture coordinates offset. The bigger this value is the better is the final quality.

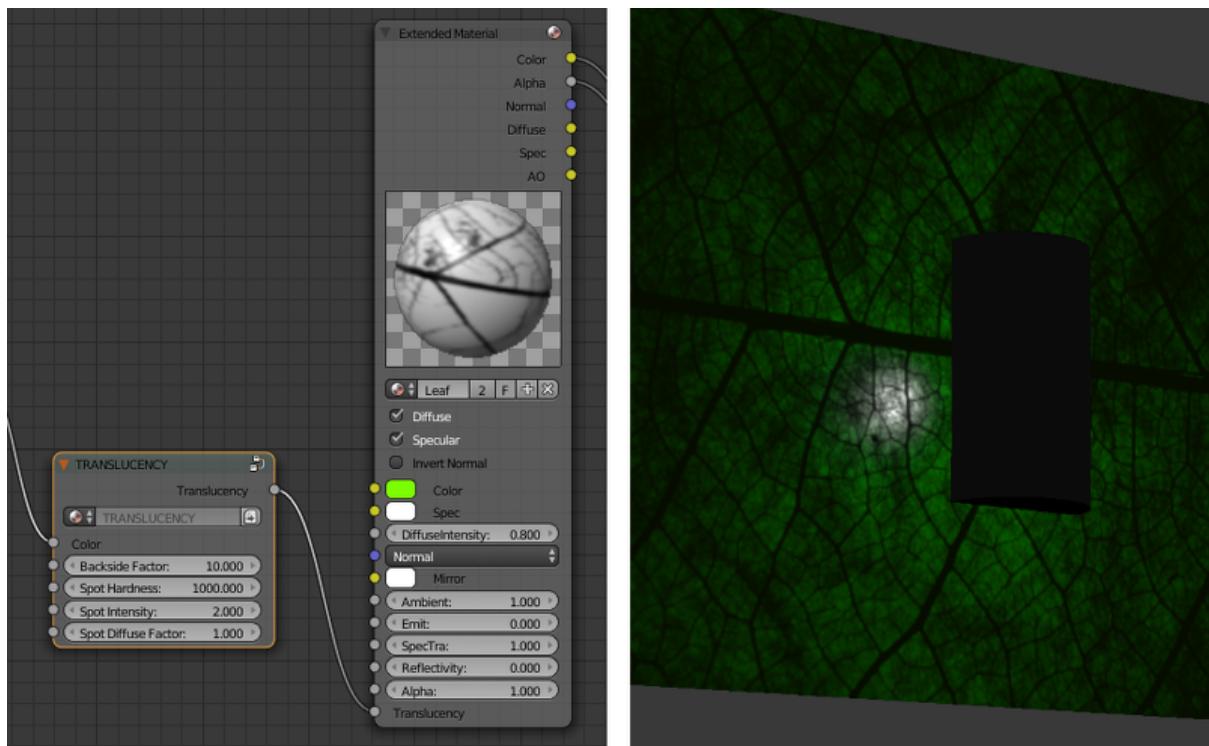
Lod Distance Maximum distance from camera at which the effect is observed.

Output Parameters

UV Resulting texture coordinates which are used as input for the texture nodes.

Translucency (B4W_TRANSLUCENCY)

The node implements a translucency effect (with respect to light sources only) for thin objects such as cloth, leaves, paper etc. The effect consists of two parts: 1) brightening of the object side which is opposite to the light source and 2) appearance of a light spot right in the light source place.



Input Parameters

Color One-channel texture which defines material heterogeneity - the white color denotes maximum translucency effect while the black color denotes its absence. White color is used by default.

Backside Factor Material color correction coefficient for the side which is opposite to the light source. It describes the color richness effect for the translucent areas.

- Backside Factor < 1 - brightening
- Backside Factor $= 1$ - no correction
- Backside Factor > 1 - darkening

The default value is 1.

Spot Hardness Light spot blurring factor. The bigger this value is the smaller is the spot and the sharper are the spot edges. The default value is 1000.

Spot Intensity Light spot intensity. The bigger this value is the brighter is the light spot.

The default value is 1.

Spot Diffuse Factor Material diffuse color influence on the light spot color.

- Spot Diffuse Factor = 0 - the light spot has the diffuse color
- Spot Diffuse Factor = 1 - the light spot color is white

The default value is 1.

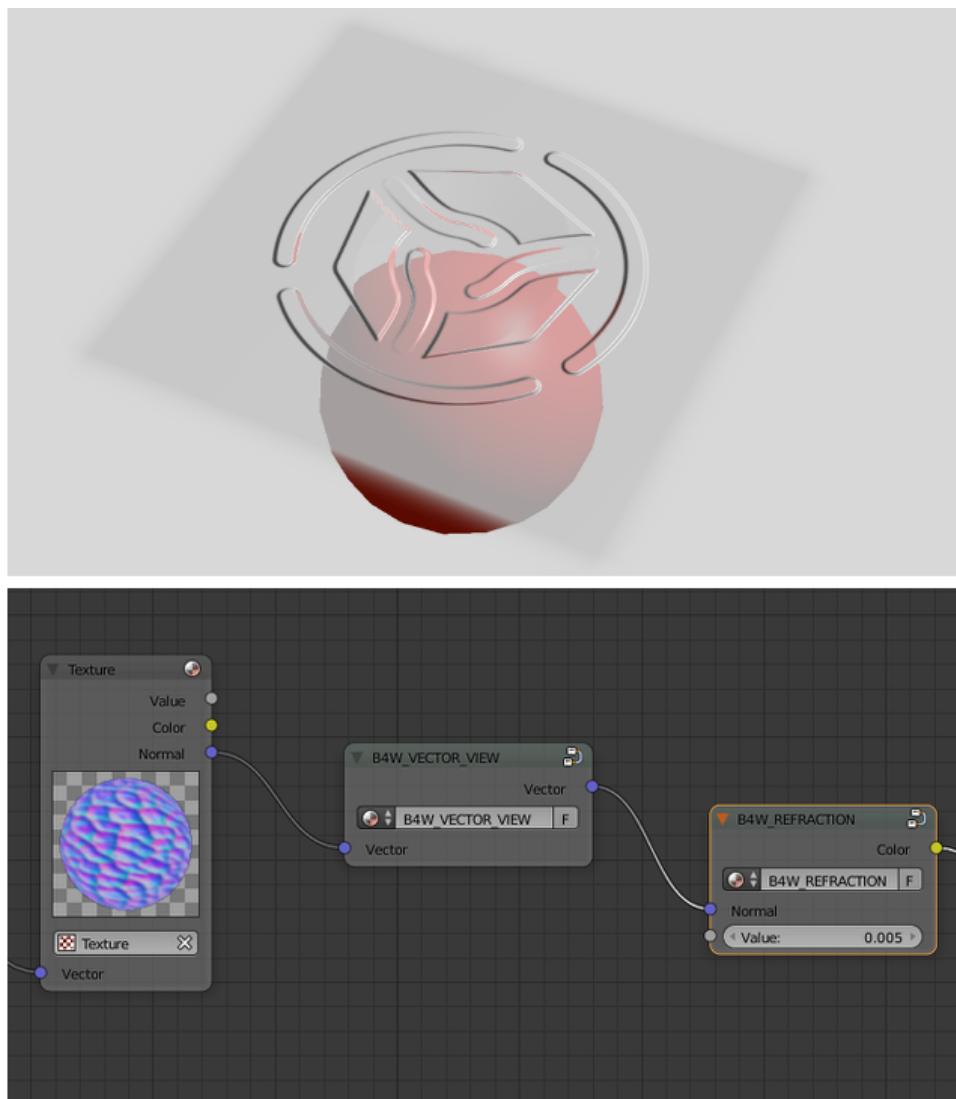
Output Parameters

Translucency The output should be connected to the Translucency input of the Extended Material node.

Note: This node can work incorrectly, if the mesh normals were edited.

Refraction (B4W_REFRACTION)

Applies refraction effect to an object. This effect works only in the Blend4Web engine and not in the Blender's viewport.



Input Parameters

Normal A normal vector in camera space for adding perturbations.

Refraction Bump Value of perturbation strength.

The default value is 0.001.

Output Parameters

Color Rendered texture behind object with perturbations.

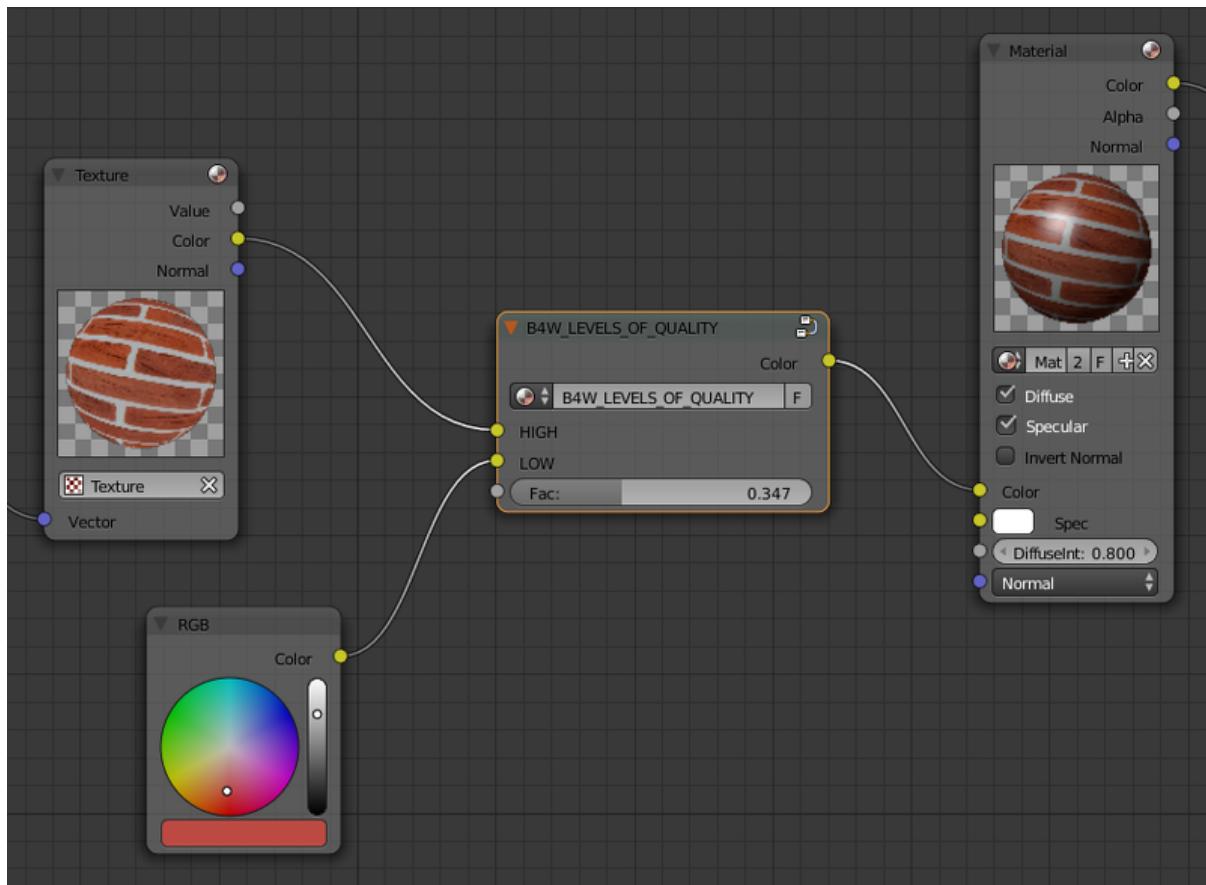
Note: It's necessary to set the Refractions option from the Render > Reflections and Refractions panel to value AUTO or ON. The object's transparency type must be set to Alpha Blend.

See also:

Transparency

Levels of Quality (B4W_LEVELS_OF_QUALITY)

Sets up the output color based on the current image quality settings. Can be used to, for example, replace complex material with more simple one if the application is running on a mobile device.



See also:

[Quality Profiles](#)

Input Parameters

HIGH Node links this parameter to Color parameter in case of high and maximum quality usage.

LOW Node links this parameter to Color parameter in case of low quality usage.

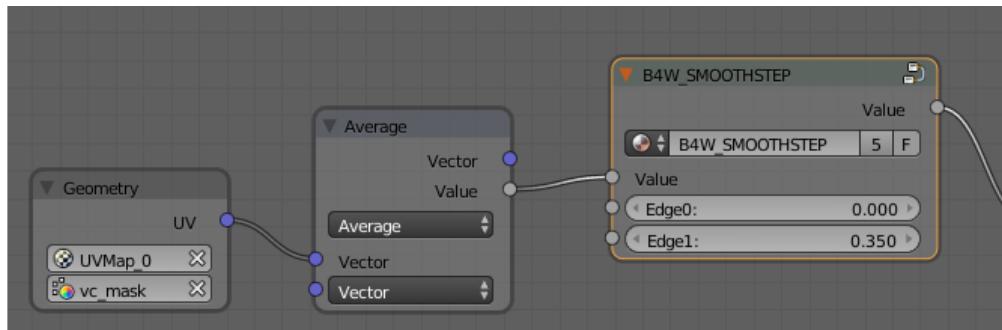
Fac This value specifies what quality setting (HIGH or LOW) will be visible in the Blender viewport. Can change from 0 to 1. If the value is lower than 0.5, the HIGH setting will be visible, if it is 0.5 or higher, the LOW setting will be visible.

Output Parameters

Color Output color.

Smoothstep (B4W_SMOOTHSTEP)

Performs smooth interpolation between two input values based on first value.



Input Parameters

Value Value which determines interpolation smoothness.

Edge0 First interpolation value.

Edge1 Second interpolation value.

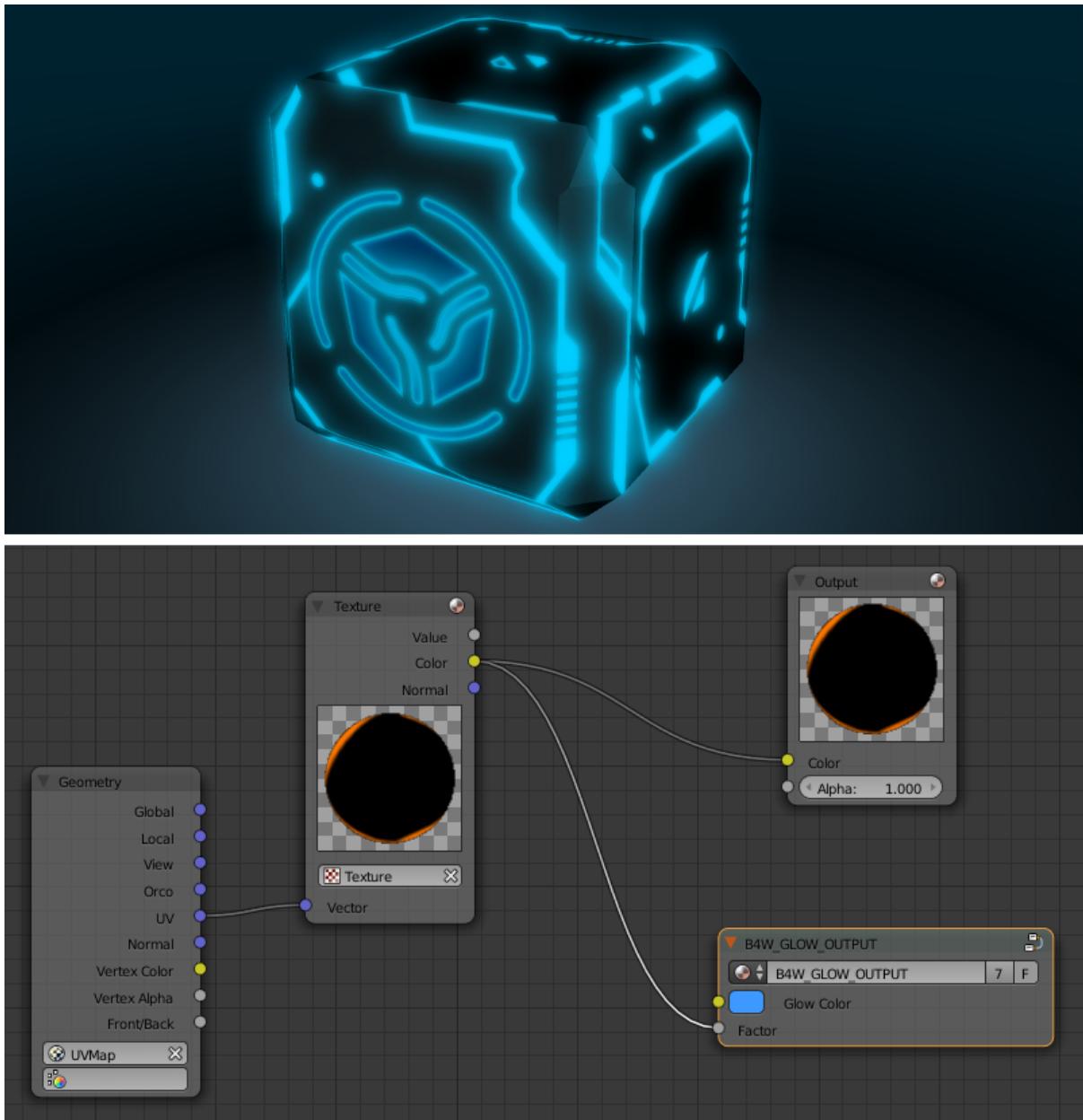
Output Parameters

Value Interpolated value.

Note: For the correct interpolation input Value had to be between Edge0 and Edge1.

Glow Output (B4W_GLOW_OUTPUT)

Applies the Glow effect to the node material. Besides the B4W_GLOW_OUTPUT node, the node material should have the Output node.



Input Parameters

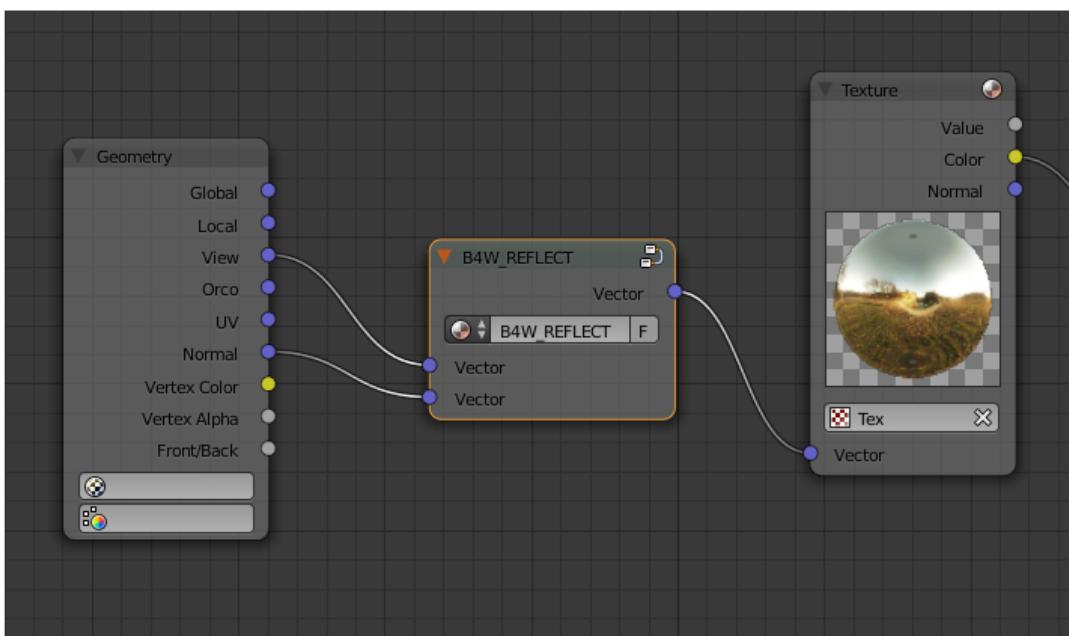
Glow Color Glow color.

Factor Glow ratio. Factor $\in [0, 1]$.

- Factor = 0 - no glow.
- Factor $\in (0, 1]$ - there is a glow, colored with Glow Color.

Reflect (B4W_REFLECT)

Calculates the reflection direction for an incident vector and a normal vector. Can be used to apply a cubemap to an object.



Input Parameters

Vector Incident vector. Should be connected to the View socket of the Geometry node.

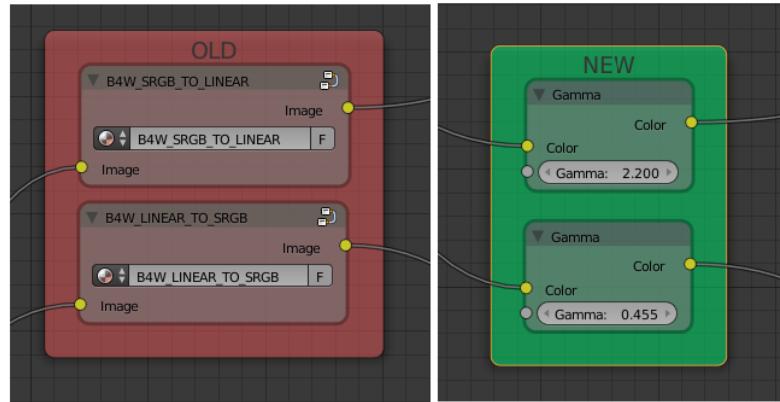
Vector Normal vector. Should be normalized in order to achieve the desired result.
Should be connected to the Normal socket of the Geometry node.

Output Parameters

Vector Reflected vector. Should be connected to the Vector socket of the Texture node that contains the cubemap.

Linear to sRGB and sRGB to Linear (Deprecated)

Converts colors from linear space to sRGB or vice versa. This function has been declared deprecated since the version 15.04. In the newer versions, the native `Gamma` node with the value of 2.200 should be used to convert color from sRGB to linear space, and the same node with the value of 0.455 to convert color from linear space to sRGB.



See also:

[Gamma in Node Materials](#)

Logic Nodes

Table of Content

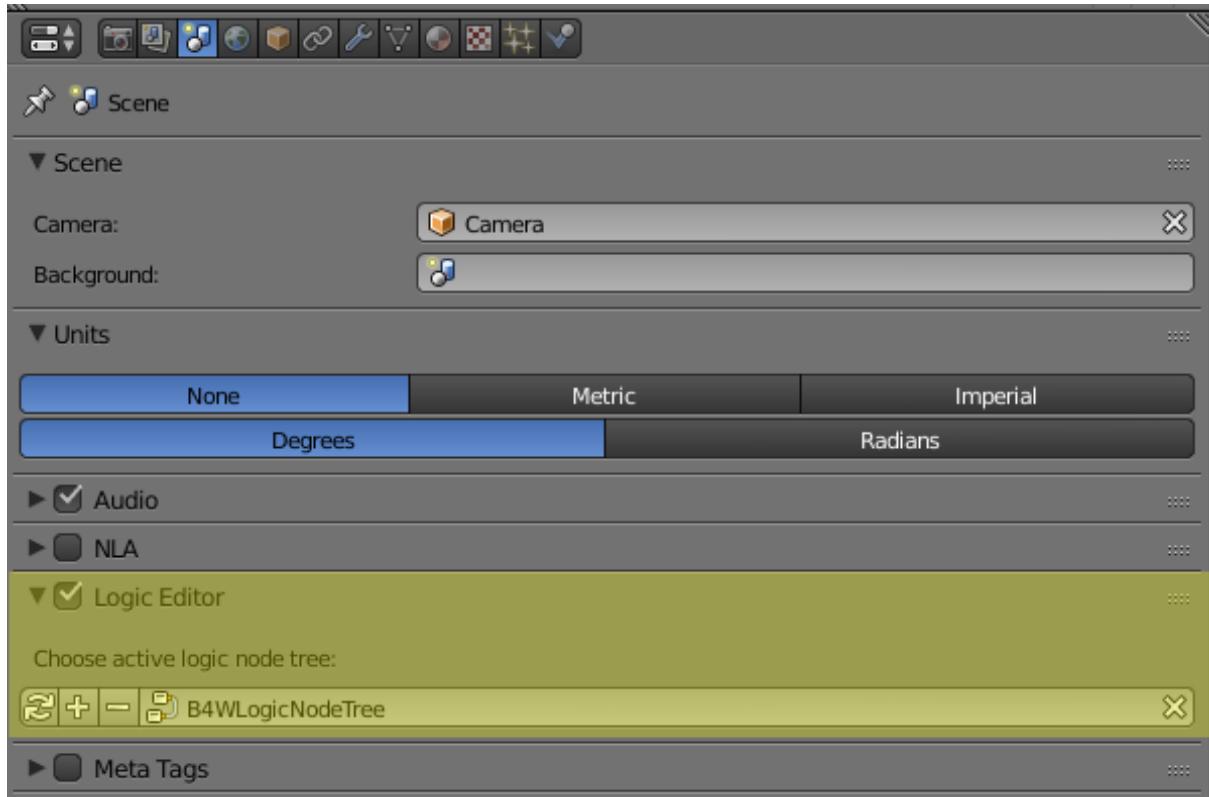
- Logic Nodes
 - Basics
 - Control Flow
 - Animation
 - Camera
 - Object
 - Operations
 - Sound
 - Network
 - Debug
 - Deprecated
 - Layout
 - Debugging

Basics

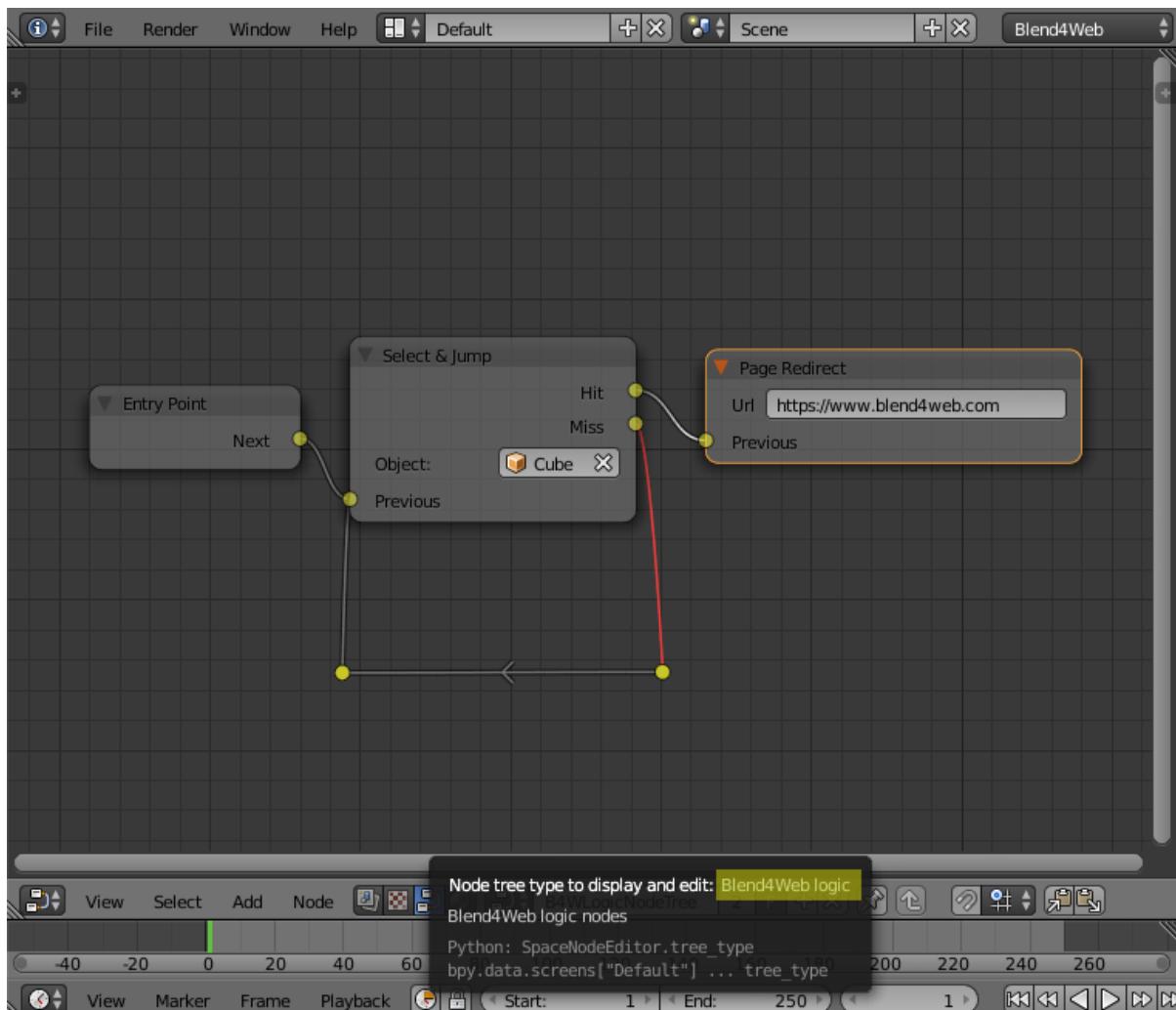
Visual programming is performed by creating logic node tree in the node editor area in Blender. These nodes can extend the scene functionality significantly without any coding.



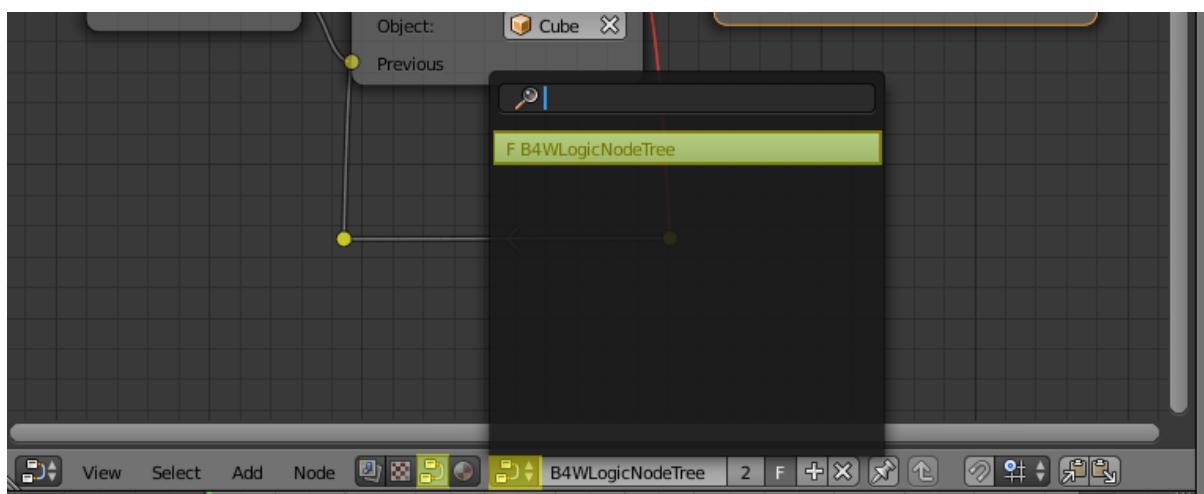
To activate logic on the given scene set the Logic Editor property in the Scene tab and append/select active node tree.



To edit logic tree go to the Node Editor area and select the Blend4Web Logic tree type:



Then select the required node tree:

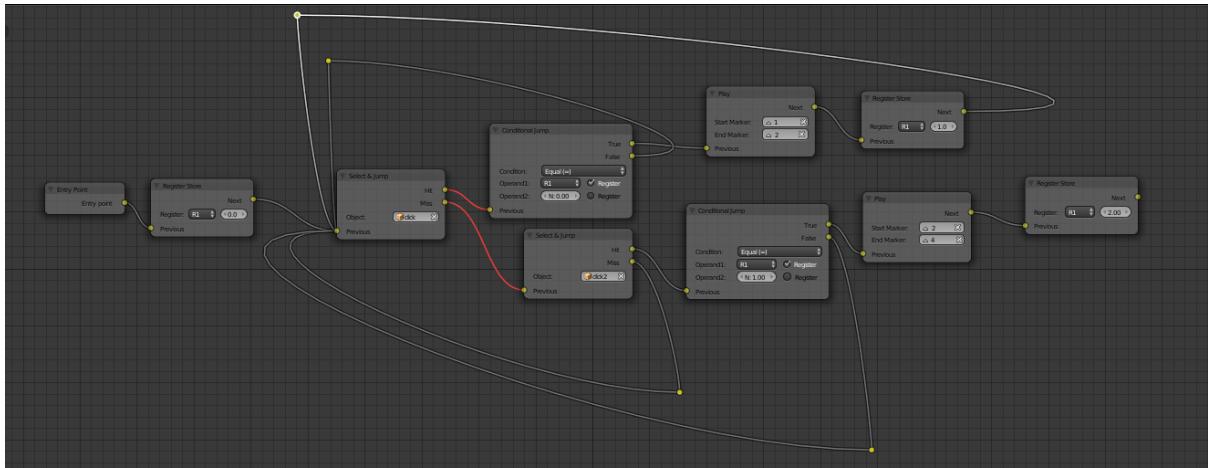


Nodes are created by using standard Blender keyboard shortcut Shift+A.

The nodes themselves are logic blocks that are executed from the Entry Point node which has no inputs and only one output. All other nodes have both inputs and outputs, and can be inserted in any place of a logic tree. The exception is Page Redirect node, which can be inserted only to the end of the node tree. The nodes which have two outputs allow branching, thus the next leave is selected based on condition specified for such nodes.

For implementing complicated logic there are variables that can have either numeric or string values. The variables can be used for storing some scene state (e.g. this can be a counter of animation playbacks, character's health points etc).

Logic Editor usage example:

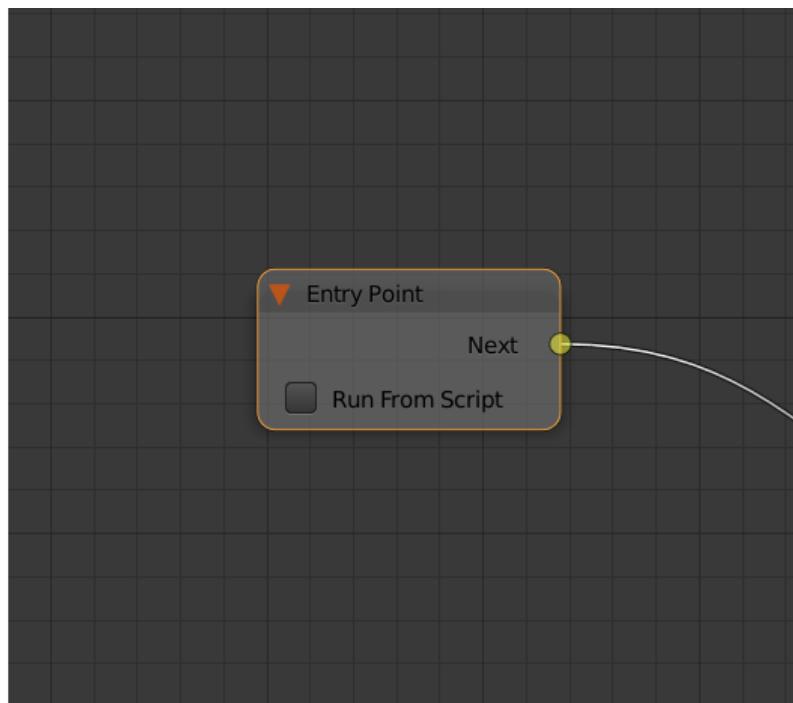


All available nodes are described below.

Control Flow

Entry Point

This is where the script starts. By using multiple entry points you can create multi-threaded applications.



Input Parameters

None.

Output Parameters

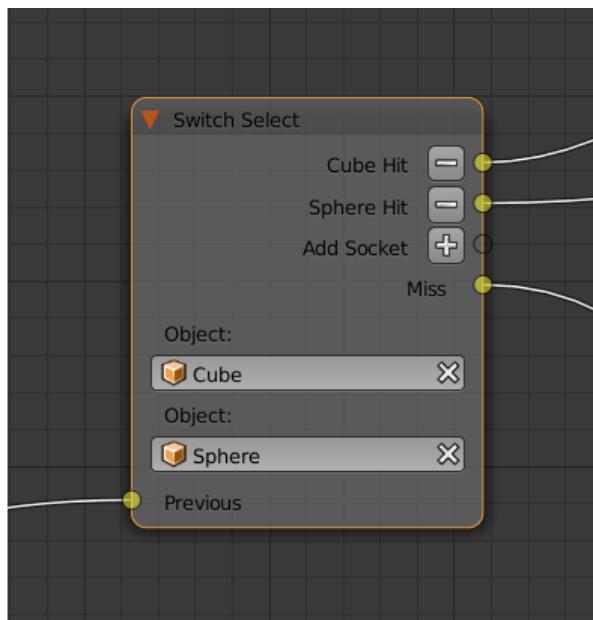
Next Next node.

Internal Parameters

Run From Script If this parameter is enabled, the entry point can be triggered via API by using the `run_entrypoint` method.

Switch Select

Can be used to trace the selection of any object from the object list.



Input Parameters

Previous Previous node.

Output Parameters

<object name> Hit This parameter will pass the control to the next node if the user selects (with a mouse or by touch) an object mentioned in the parameter's name. The Switch Select node has one such parameter by default, but you can add new ones and remove existing ones (the node can even have no such parameters).

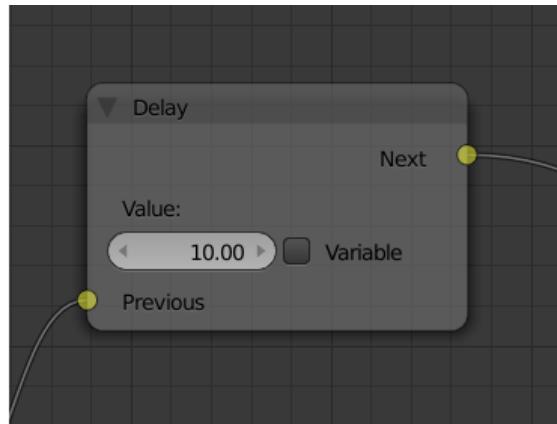
Miss This parameter will pass the control to the next node if the user hasn't selected any of the objects specified in the Switch Select node.

Internal Parameters

Object One of the objects that the user can select. These parameters are automatically created and deleted when you create or delete a Hit parameter. The number of such parameters is always equal to the number of the Hit parameters.

Delay

Make a delay before going to the next node.



Input Parameters

Previous Previous node.

Output Parameters

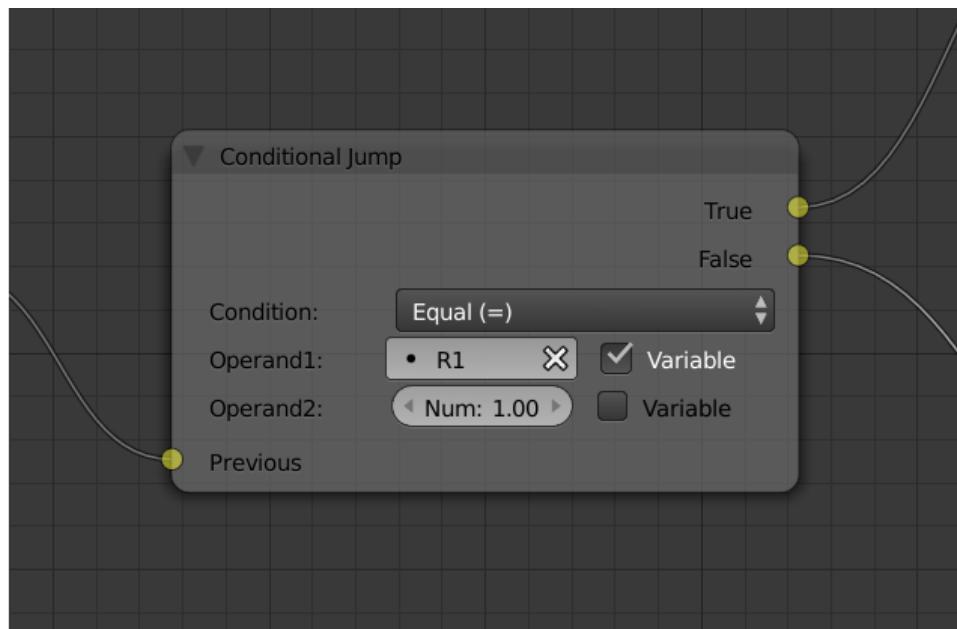
Next Next node.

Internal Parameters

Value Time (in seconds) that will pass before the activation of the next node. Set to zero by default. Can be set manually or through a variable (if the Variable parameter is enabled).

Conditional Jump

Go to the specified node if the certain condition is met. The parameters (operands) can also be variables that are activated using the corresponding switches.



Input Parameters

Previous Previous node.

Output Parameters

True Condition is true.

False Condition is false.

Internal Parameters

Condition Logical condition. Can have one of the following types:

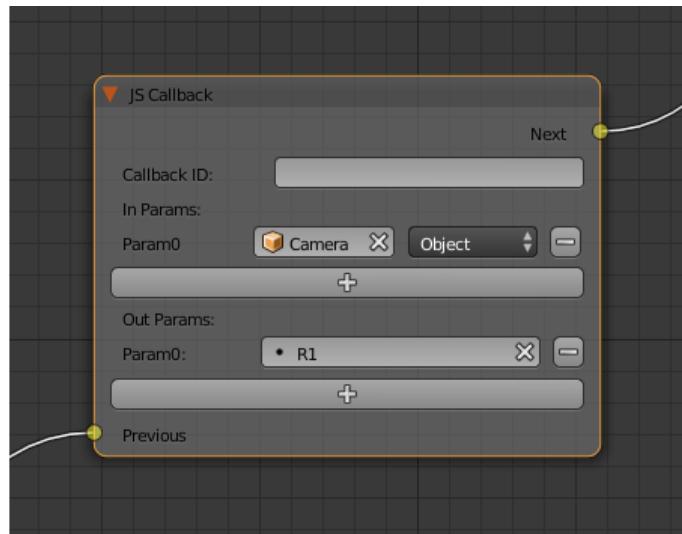
- Equal - first operand is equal to the second.
- Not Equal - first operand is not equal to the second.
- Less Than - first operand is less than the second.
- Greater Than - first operand is greater than the second.
- Less Than Or Equal - first operand is less than or equal to the second.
- Greater Than Or Equal - first operand is greater than or equal to the second.

Operand1 First operand of the logical condition. Should have a numeric value. Can be specified in the node or can be a link to one of the variables.

Operand2 Second operand of the logical condition. Works the same way as the first.

JS Callback

Can be used to call custom JavaScript callback defined in your Blend4Web application.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

Callback ID The ID of a JavaScript function that will be called by the node.

In Params A list of the input parameters of the function. Each parameter can be either a variable or a link to a scene object. The number of the input parameters can be adjusted. By default, this list is empty.

Input parameters are transferred to the callback function as an array that servers as the first argument of the function.

Param <param_number> Specifies an input parameter. This parameter can be a variable (R1 by default) or a link to a scene object, depending on the value of the Type parameter (each one of these parameters always has a corresponding Type parameter).

Type The type of the corresponding input parameter. It can have one of the two values: Variable (in this case, one of the variables will be used as the parameter) and Object (a link to a scene object).

Out Params A list of the output parameters. Empty by default. The number of the output parameters can be adjusted.

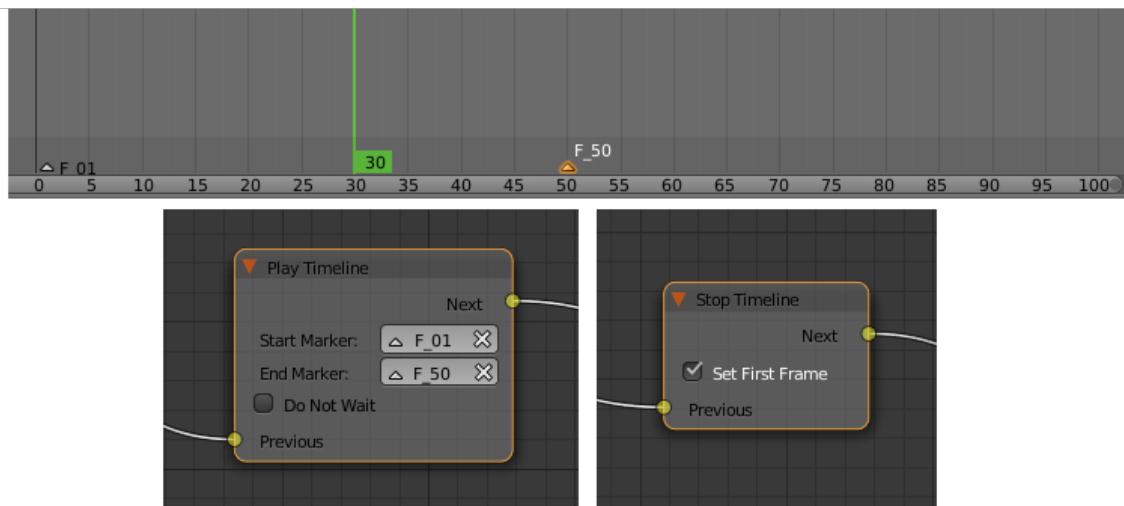
An array that consists of the output parameters serves as the second argument of the callback function.

Param <param_number> Specifies one of the variables that will be used as an output parameter. By default, R1 variable is used.

Animation

Play Timeline and Stop Timeline

Can be used to control NLA animations. The Play Timeline node plays NLA fragment starting with a frame specified by the marker. Animation plays until next marker is encountered, or to the end of the scene's timeline. After that, control passes on to the next node. The Stop Timeline node stops the playback.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

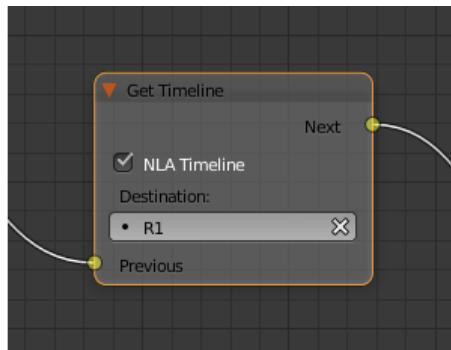
Internal Parameters

Start Marker First frame of the animation. If not specified, an animation plays from the start of the timeline and may not work correctly.

End Marker Last frame of the animation. If not specified, an animation plays to the end of the timeline and may not work correctly.

Get Timeline

This node can be used to get the current frame of an NLA animation or a timeline.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

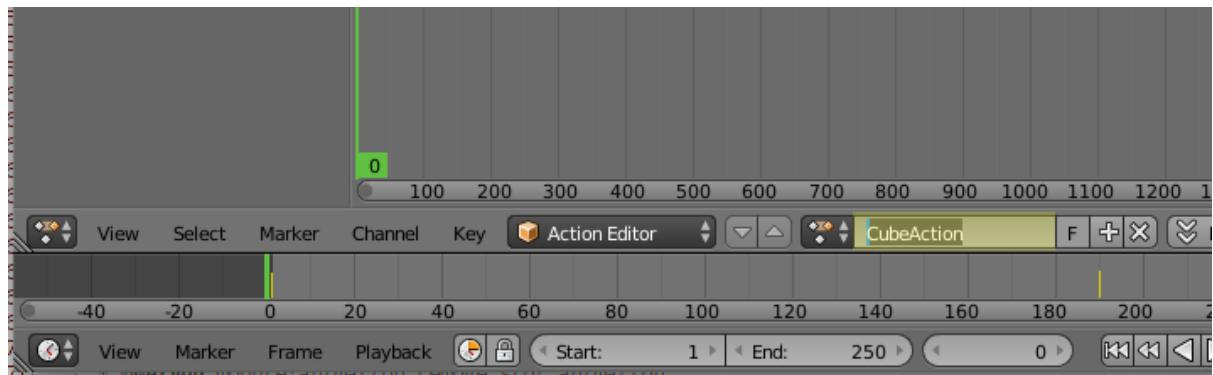
NLA Timeline If this parameter is enabled, the node will return the current frame of an NLA animation. If it is disabled, the node will return the current frame of the timeline. Enabled by default.

Destination Specifies a variable to store the number of the current frame. Set to R1 by default.

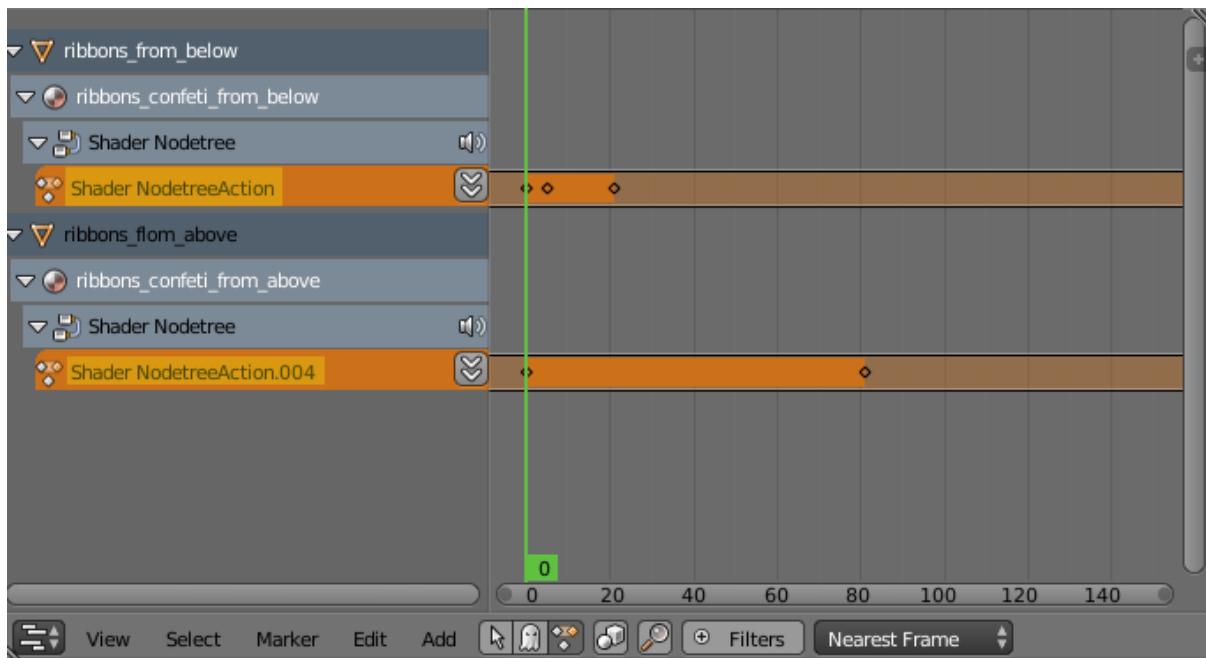
Play Animation

Can be used to play object's animation. An animation can be one of the following types:

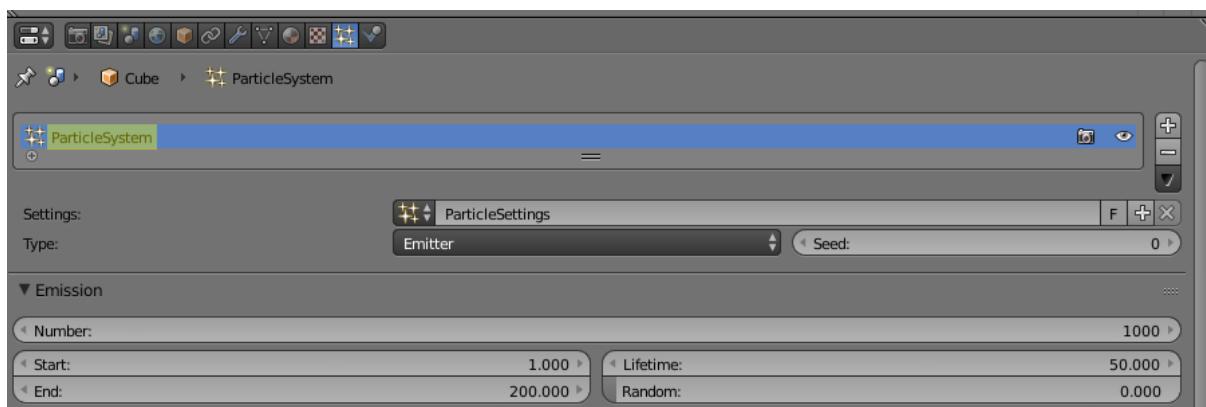
Regular Action:



Shader Action:



Particle system:



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

Object Name of the object, animation of which will be played.

Anim. Name Name of an animation to play. If not specified, the entire timeline will be played.

Behavior Specifies animation behavior. Can have one of the following values:

- Finish Stop - animation will be played once.
- Finish Reset - animation will be played once, then the object will return to the starting point.
- Loop - animation will be playing repeatedly until it is stopped by the Stop Animation node.

Do Not Wait If this parameter is enabled, the Play Animation node will pass the control to the next node on starting the animation playback. If it isn't, the control will be passed to the next node only after playback is finished.

Stop Animation

Can be used to stop an object's animation.

Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

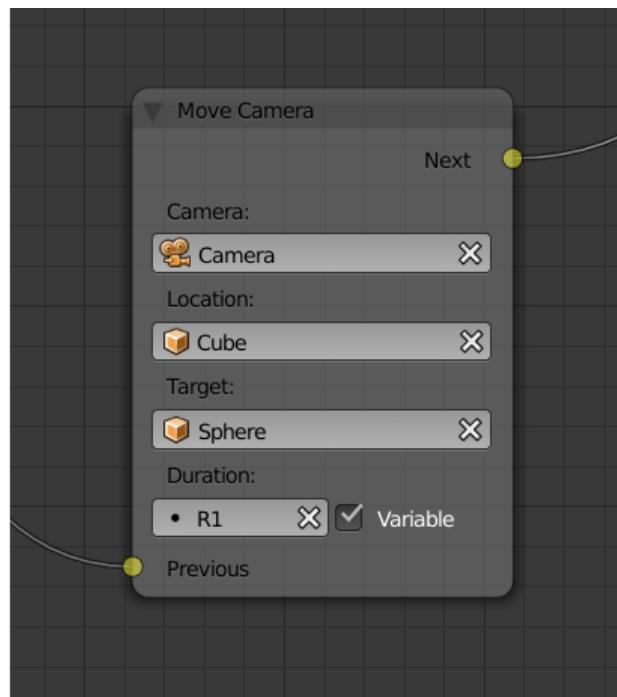
Internal Parameters

Set First Frame Go back to the first frame after the animation has been stopped.

Camera

Move Camera

Can be used to move the camera, including smooth interpolation of its position.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

Camera A camera that will be moved.

Location An object to which the camera will move. The camera's coordinates will be the same as the object's after the movement is finished.

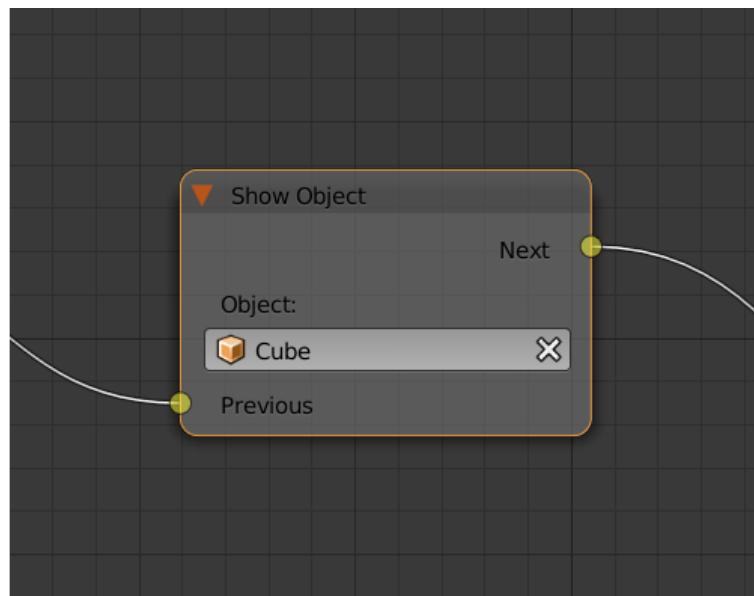
Target The camera will point in the direction of this object after being moved.

Duration Time (in seconds) that the camera will spend being moved to a new location. Set to zero by default (and in this case the camera doesn't actually move, it simply changes its position). It can be specified manually or as a link to a variable (if the Variable parameter is enabled).

Object

Show Object

Can be used to show 3D objects.



Input Parameters

Previous Previous node.

Output Parameters

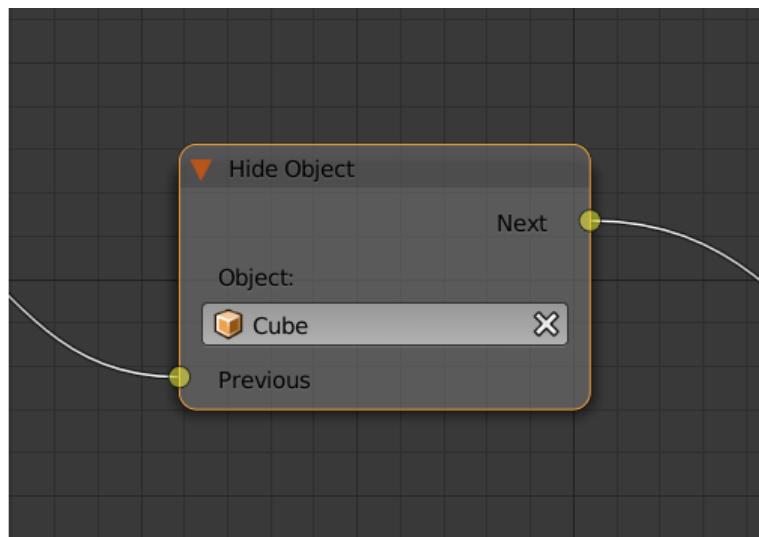
Next Next node.

Internal Parameters

Object An object to show.

Hide Object

Can be used to hide 3D objects.



Input Parameters

Previous Previous node.

Output Parameters

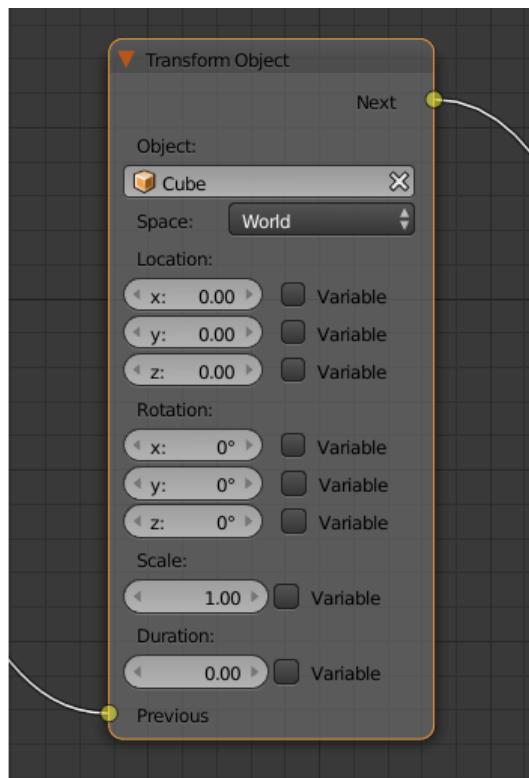
Next Next node.

Internal Parameters

Object An object to hide.

Transform Object

Can be used to transform object's location, size and rotation angles.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

Object An object that needs to be translated.

Space This parameter defines the coordinate space that will be used to transform the object. It can have one of the following values:

- World - global coordinate space.
- Parent - local coordinate system of the parent of the object specified by the Object parameter. Parent object's origin point is used as the center of coordinates, while its angles of rotation define the directions of the coordinate axes.
- Local - local coordinate space of the selected object. Similar to the Parent coordinate space, but in this case, the origin point of the object itself is used as the origin of coordinates.

Set to World by default.

Location How the object will move along the X, Y and Z axes. By default, all three parameters are set to zero. Values can be specified in the node itself or through the variables (if the Variable option is enabled).

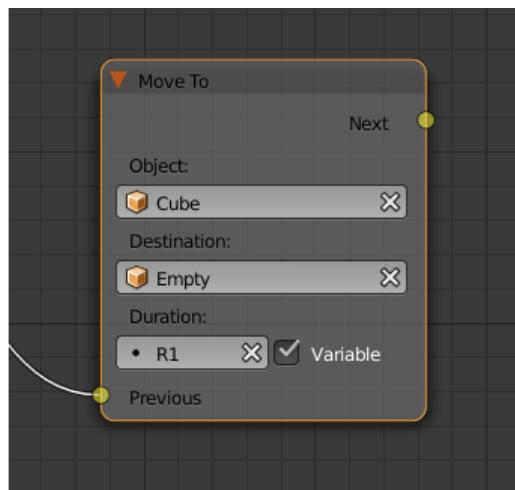
Rotation Object's rotation around the X, Y and Z axes. All three values are set to zero by default. Can be specified directly in the node or through the variables (if the Variable option is enabled).

Scale Object's size. Can be specified directly or through a variable (if the Variable parameter is enabled). Set to 1 by default.

Duration Time (in seconds) that the transformation will take. It can be specified both directly or with a variable (to do this, the Variable parameter should be enabled). Set to zero by default.

Move To

Can be used to move objects.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

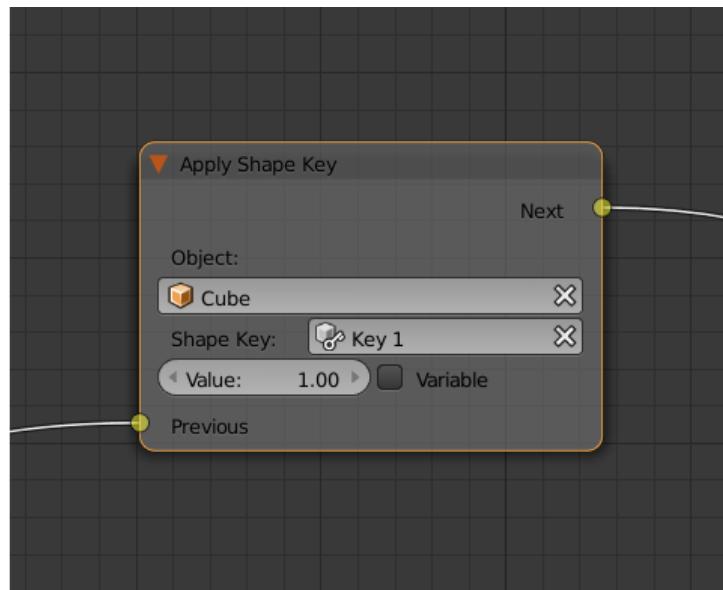
Object An object that you need to move.

Destination A target (another object or a light source, camera or anything else) to which the selected object will move. The object's coordinates will be the same as the target's after the movement is finished.

Duration Time (in seconds) that the object will spend moving to the new location. By default, this parameter is set to zero (and in this case, the object doesn't actually move, it just changes its position in a moment). It can be set manually or with a variable (available only if the Variable parameter is enabled).

Apply Shape Key

Set the Shape Key factor.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

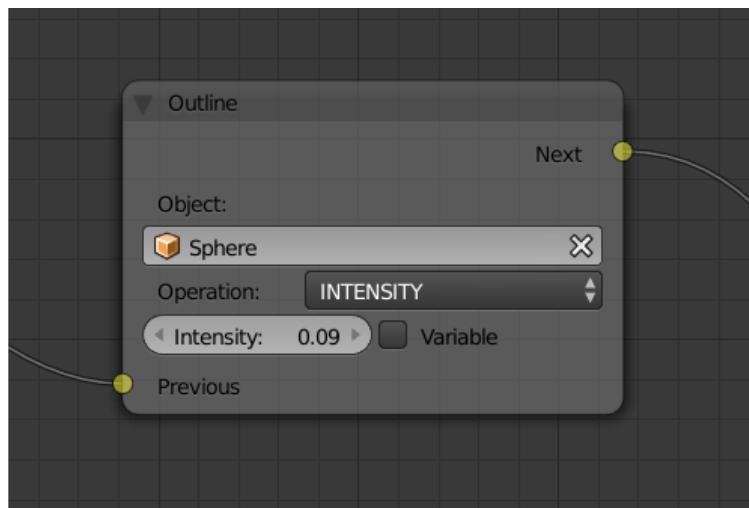
Object An object that needs to be transformed.

Shape Key Shape key that will be applied to the object.

Value How much the shape key will influence the object. This value can be set directly in the node or using a variable. The value should be between 0 and 1.

Outline

Controls object outlining effect.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

Object Any changes of the outline effect will be applied only to an object specified by this parameter.

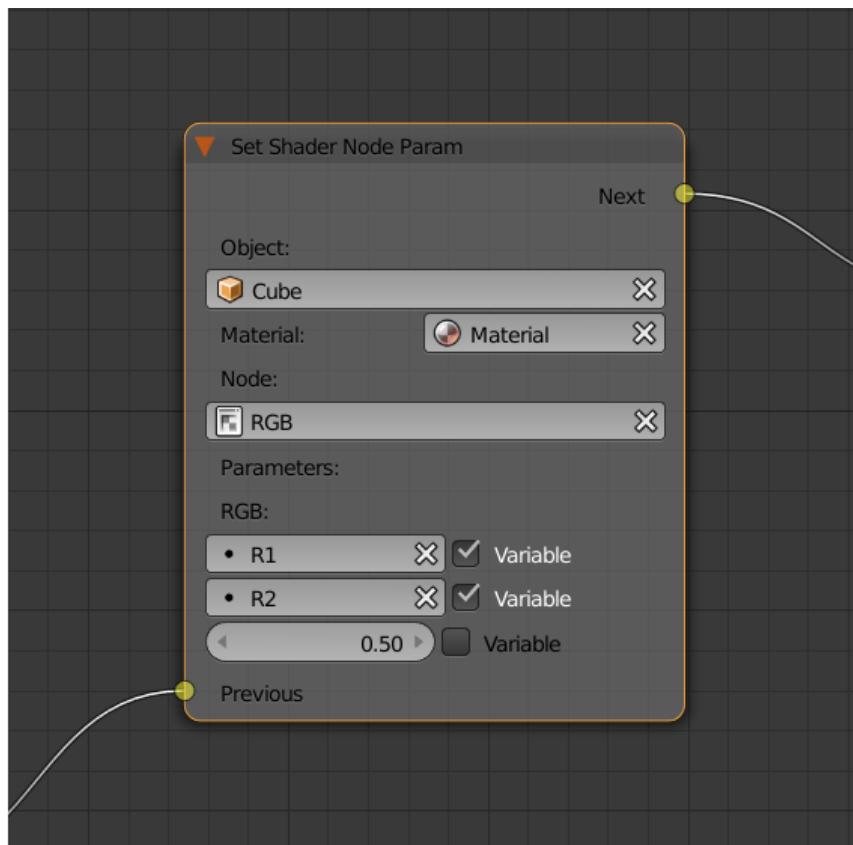
Operation Specifies an operation that will be done to the object's outline. This parameter can have one of the following values:

- PLAY enables outline animation
- STOP disables it
- INTENSITY can be used to set intensity of the object's outline

Intensity Outline intensity. This parameter is only available if the Operation parameter is set to INTENSITY. The value can be set manually or via variable (if the Variable parameter is enabled).

Set Shader Node Param

Can be used to set the value for the shader node. Currently, only Value and RGB nodes are supported.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

Object An object that has material that needs to be edited.

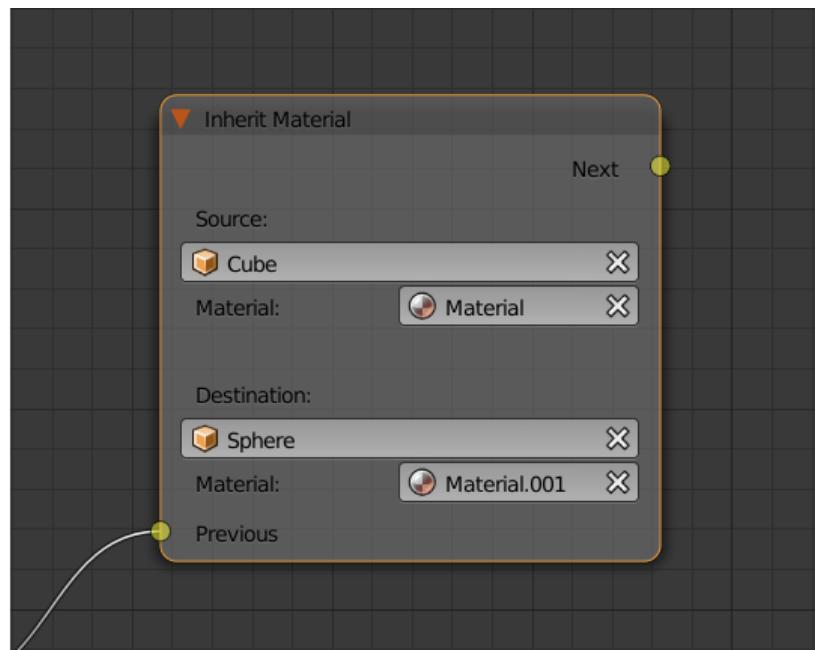
Material Material that needs to be edited. It should use nodes.

Node A node that has parameters that can be changed. For now, only Value and RGB nodes are supported.

Parameters Editable parameters of the selected node. They can be set in the node itself or through the variables (if the Variable parameter is enabled).

Inherit Material

Copy attributes from one material to another.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

Source An object that has a material that will be inherited.

Material The material to inherit.

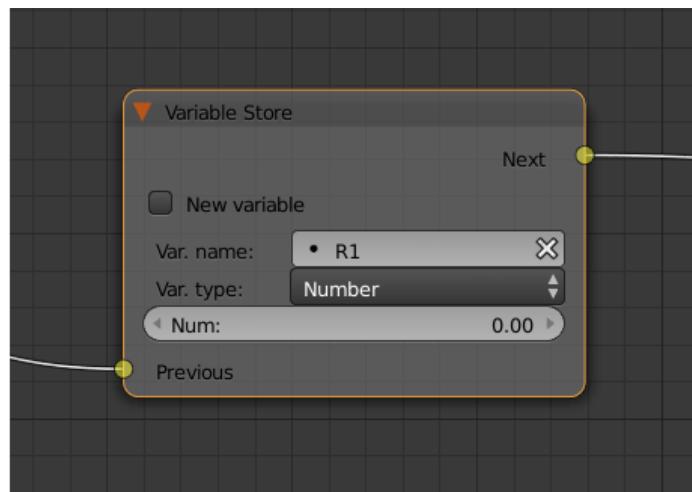
Destination An object that will inherit selected material.

Material The material that will be replaced by the inherited one.

Operations

Variable Store

Saves numerical or string value to a variable.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

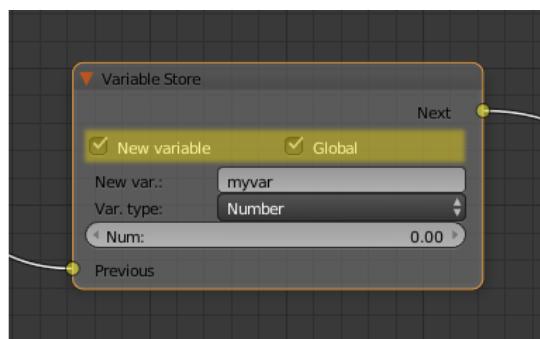
Internal Parameters

Var. name. Name of the variable. Can be selected from the list of variables or specified manually (if the New variable parameter is enabled).

Var. type Variable's type. This parameter can have one of two values: Number (for numerical variables) and String (for string variables).

New Variable If this parameter is enabled, you can manually input a variable's name and not just select one of the variables. This can be used to transfer the data between the application and the server.

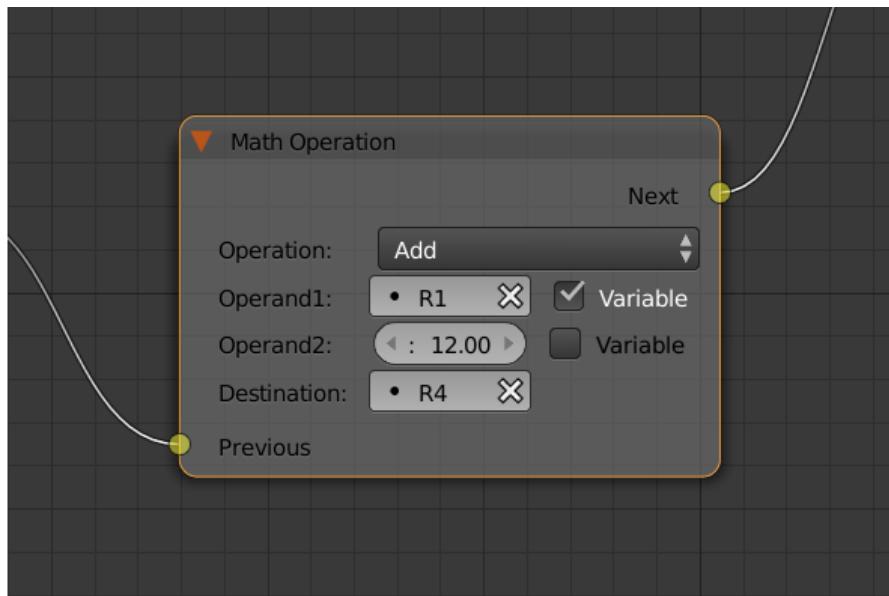
Global Enabling this parameter makes the variable global. Available only if the New Variable parameter has been enabled.



Num./Str. Numeric or string (depending on the Var. type parameter value) value of the variable.

Math Operation

Perform a math operation and store the result in the variables. Any of parameters (operands) can be either a numeric value or a variables.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

Operation Mathematical operation. Can have one of the following types:

- Random generates random value greater than the first operand and less than the second.
- Add sums the operands.
- Multiply multiplies the operands.
- Subtract subtracts the second operand from the first.
- Divide divides first operand by the second.

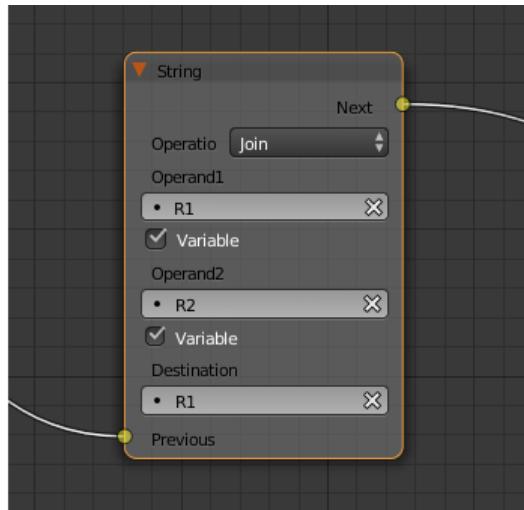
Operand1 First operand. It can be specified in the node or it can be a link to one of the variables (if the Variable parameter is enabled).

Operand2 Second operand. Works the same way as the first.

Destination The result of the operation will be saved in the variable specified by this parameter.

String Operation

Can be used to perform an operation with two strings and save the result to a variable.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

Operation An operation that you need to perform with two strings, which can have one of the following values:

- Join - joins two strings into one.
- Find - writes the index of the first occurrence of the second string in the first string to the variable. If there is no occurrences, the value of -1 will be written. It should be noted that the first symbol of a string has an index of 0, not 1.
- Replace replaces first occurrence of the second string in the first string with the third one.
- Split splits the first string in two using the first occurrence of the second string as a splitting mark.

- Compare compares two strings. For this operation, you need to specify a logical condition. If this condition is met, a value of 1 will be outputted to the Destination variable, if it isn't, zero will be outputted.

Condition A logical condition to compare two strings. This parameter is available only if the Operation parameter is set to Compare. Works the same way as the Condition parameter of the Conditional Jump node.

Operand1 The first string. Can be specified in the node itself or with a variable.

Operand2 The second string. Works the same way as the first.

Operand3 This parameter is available only if the Operation parameter is set to Replace. Can be used to specify the third string, which will replace the first occurrence of the second one.

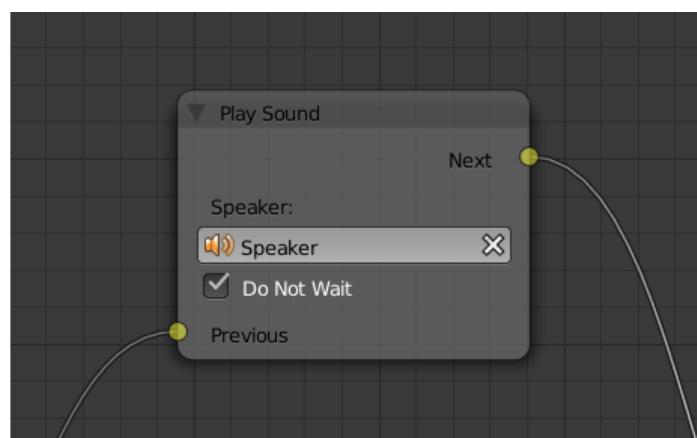
Destination A variable to save the operation's result.

Destination2 This parameter is available only if the Operation parameter is set to Split. Specifies the variable to save the second half of the string that has been split (the first half will be saved to the variable specified by the Destination parameter).

Sound

Play Sound

Can be used to play speaker's sound.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

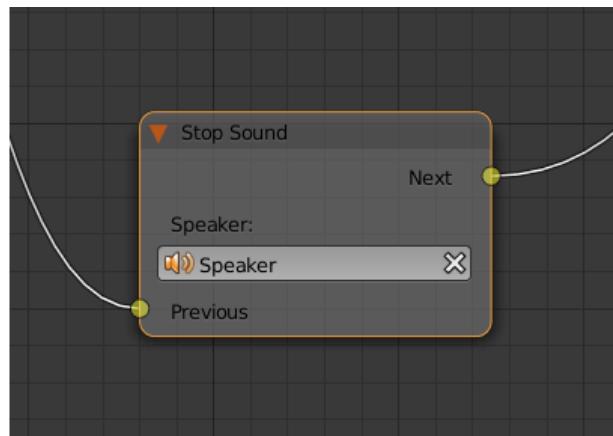
Internal Parameters

Speaker A speaker that will be enabled.

Do Not Wait If this parameter is enabled, the control will pass to the next node right after sound playback starts. If it isn't enabled, the control will pass only when the playback is finished.

Stop Sound

Can be used to stop speaker's sound.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

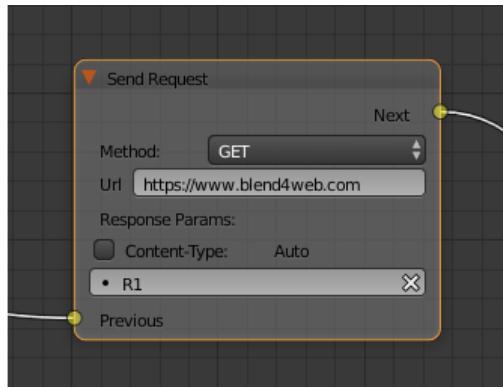
Internal Parameters

Speaker A speaker that will be turned off.

Network

Send Request

Send an HTTP GET request to the specified URL and save the respond's fields a variable.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

Method Method of the request. Can have one of two values:

- GET is used to request data from the server.
- POST is used to send data to the server.

Set to GET by default.

Url A web address to send request to. Set to “<https://www.blend4web.com>” by default.

Response Params Specifies the variable to save the data received from the server.

Note: The data received from the server should look like this:

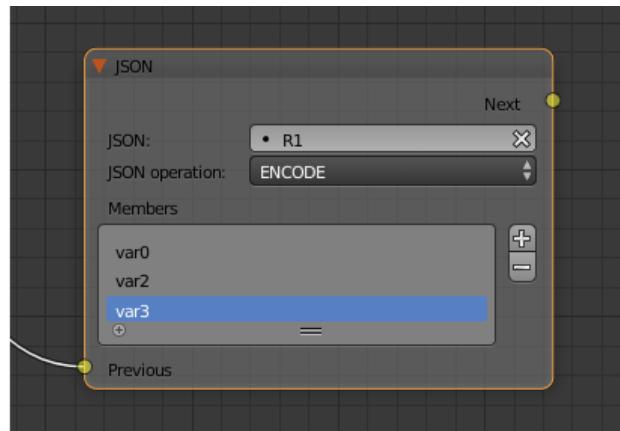
```
{"var0": 1,
"var1": 10,
"var2": 144,
...}
```

Content-Type Indicates the media type of the message content. Consists of a type and a subtype, for example: text/plain. Set to Auto by default. Can be used to reassign the title of the HTTP request.

Request Params Specifies the variable that contains a JSON object that will be sent to the server. Available only if the Method parameter is set to POST. Default value is R1.

JSON

This node can be used to encode and decode complex JSON objects.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

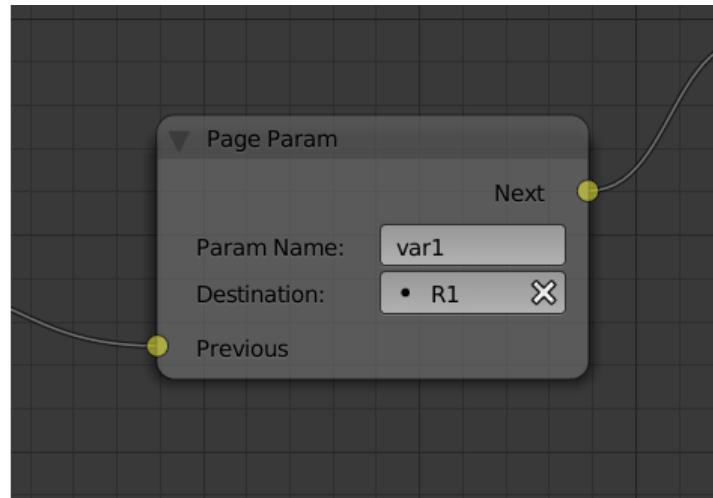
JSON A link to a JSON object that you need to decode or encode. Set to R1 by default.

JSON Operation An operation you need to perform with the JSON object specified by the JSON parameter. Can have one of two values: ENCODE to encode the JSON object and PARSE to decode it. Set to ENCODE by default.

Members A list of variables that will be used to either store the decoded data or to encode a JSON object (depending on the value of the JSON Operation parameter). The variables always have names like var0, var1 and so on, and their quantity can be adjusted.

Page Param

Allows to store any web page parameter in a given variable.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

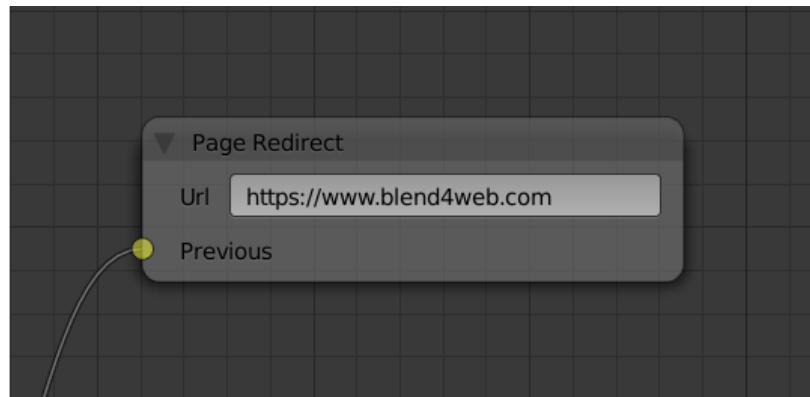
Internal Parameters

Param Name The name of the web page parameter.

Destination A variable that will be used to save the parameter.

Page Redirect

Can be used to redirect the browser to another page. This node always marks the end of the node tree and doesn't have any output parameters.



Input Parameters

Previous Previous node.

Output Parameters

None.

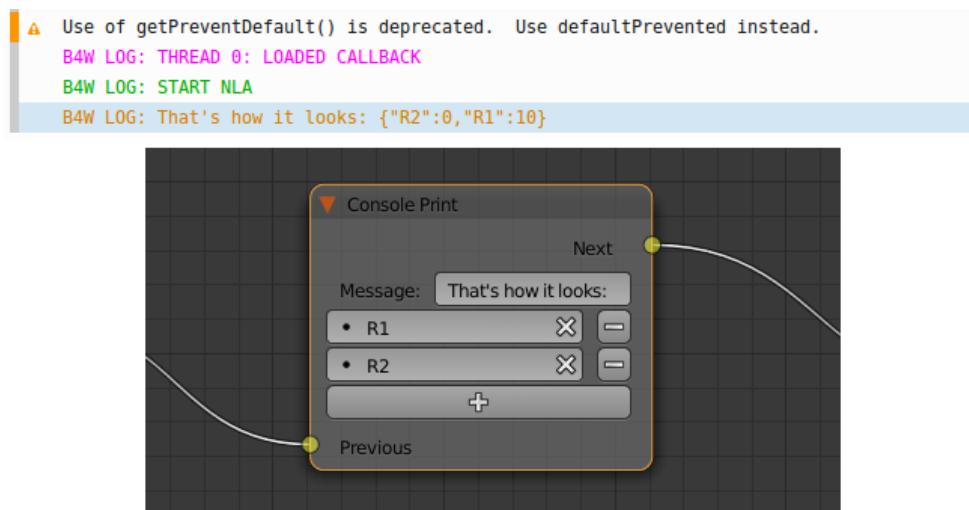
Internal Parameters

Url Web address of a page that will be opened. Set to “<https://www.blend4web.com>” by default.

Debug

Console Print

This node prints variables's values and additional text to the web browser console. It can be used for debug purposes.



Input Parameters

Previous Previous node.

Output Parameters

Next Next node.

Internal Parameters

Message A message that will be printed to the console along with the values.

<variable name> A variable that will be printed to the console. By default, a Console Print node has one such parameter, but you can add new and delete existing ones (the node might not even have such parameters at all).

Deprecated

Select (Deprecated)

Note: Deprecated! Isn't recommended to use. Instead, using a Switch Select node is recommended.

It is similar to the Select & Play node, except the transition happens instead of animation. This function allows to implement a complicated logic because in this case there is a possibility to identify the user selection results.

Select & Play Timeline (Deprecated)

Note: Deprecated! Isn't recommended to use. Instead, using a combination of Switch Select and Play Timeline nodes is recommended.

Wait until the user selects an object (on desktops - with a mouse click, on mobile devices - with a touch). If the object, which is specified in this node, is selected - then start the animation similar to the Play Timeline node. If any other object is selected - then immediately transfer control to the next node.

Select & Play Animation (Deprecated)

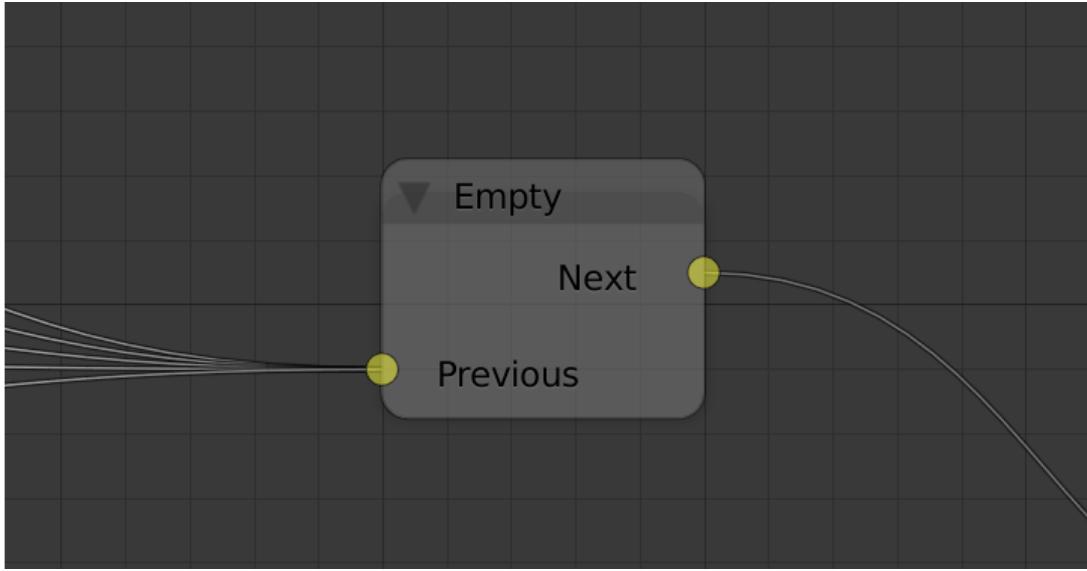
Note: Deprecated! Isn't recommended to use. Instead, using a combination of Switch Select and Play Timeline nodes is recommended.

Wait until the user selects an object (on desktops - with a mouse click, on mobile devices - with a touch). If the object, which is specified in this node, is selected - then start the animation similar to the Play Animation node. If any other object is selected - then immediately transfer control to the next node.

Layout

Empty

This is a simple pass-through node that does not perform any operations on its own. It can be used to combine several logic threads into one or simply to make the logic node setup easier to read and understand.



Input Parameters

Previous Previous node (or several nodes).

Output Parameters

Next Next node.

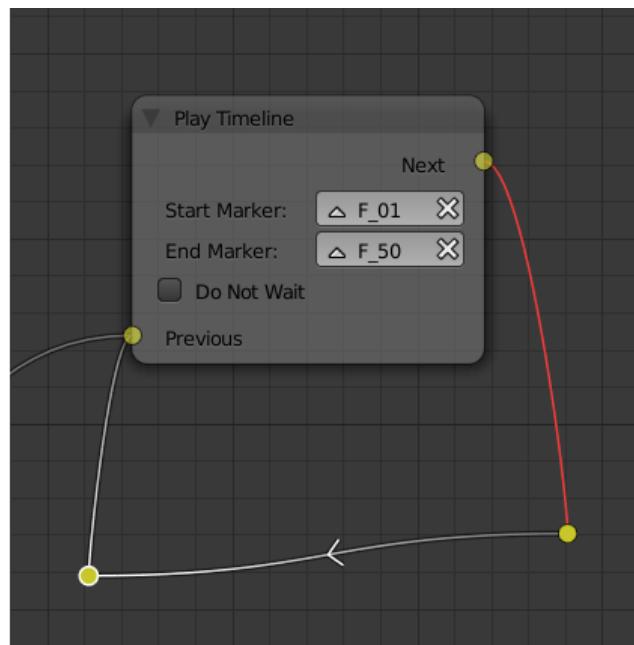
Internal Parameters

None.

Reroute

Logic Editor also has Reroute elements, the nodes that don't do anything aside from passing the control to the next node or to the next Reroute element. Such elements can be used to create cyclic structures or to make the node tree easier to read and understand.

Unlike the Empty node, Reroute element can only handle a single logic thread.



Note: Output parameter can't be connected to the same node's input parameter. If you need to do this (to make a cycle, for example), you should use Reroute elements.

Debugging

For debugging purposes some nodes inside logic tree can be muted. To do that select the required node and press the M key. Muted nodes are not evaluated and simply pass control to the next ones. If the muted node has two outputs the execution continues from the output with negative result (Miss, False).

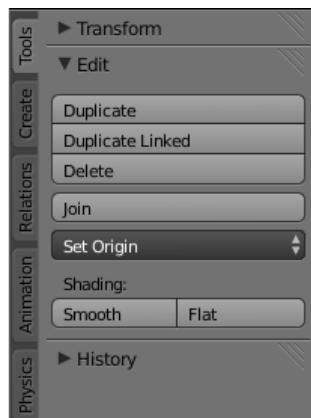
Lighting, Shadows and Background

Table of Contents

- Lighting, Shadows and Background
 - Shading Types
 - Lighting with Light Sources
 - * Light source types
 - * Light source setup
 - Environment Lighting (Ambient)
 - * Activation
 - * Setup
 - * Environment map method
 - Shadows
 - * Activation
 - * Setup
 - Background

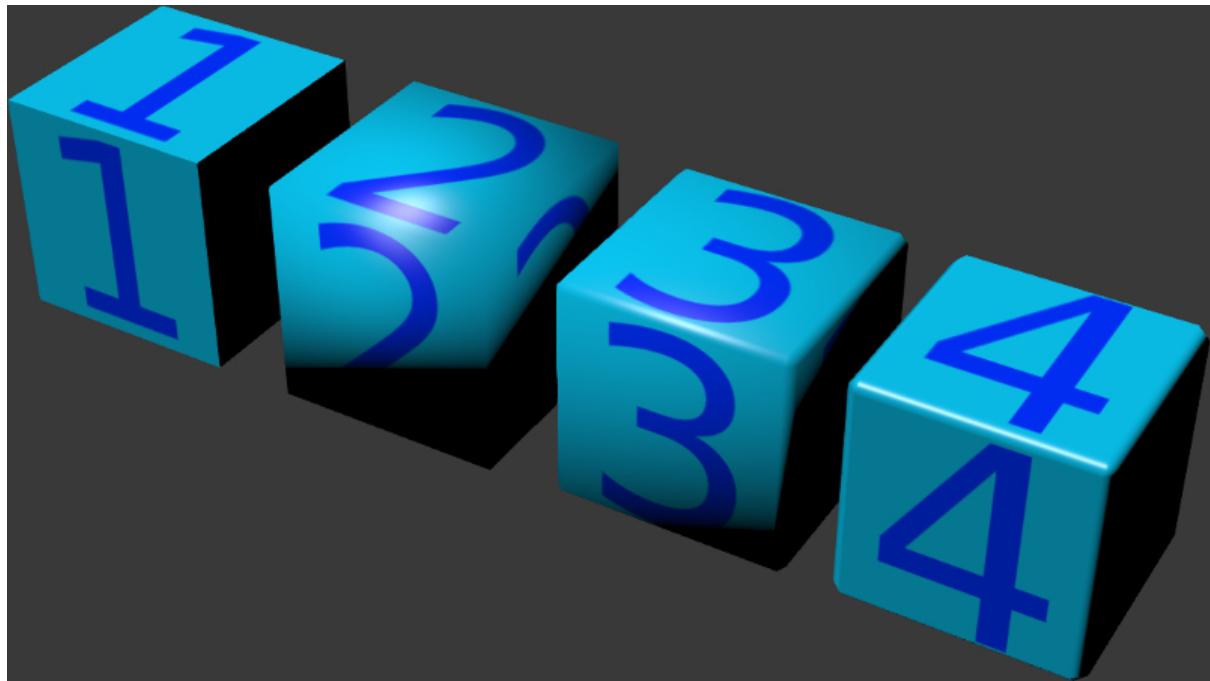
Shading Types

Lighting (shading) depends on the direction of normal vectors. The standard Blender's shading types are supported: Shading: Flat (face normals are used), Shading: Smooth (interpolated vertex normals are used) and their combinations.



If the required effect is impossible to achieve with the standard tools, you can use the [normals editor](#).

The result of applying different shading types and using the normals editor:



1. Flat Shading
2. Smooth Shading
3. Smooth Shading + bevel
4. Smooth Shading + bevel + editing normals

Lighting with Light Sources

A scene can have multiple (but not less than one) light sources of different types.

Light source types

The following light source types are supported:

Point Light propagates from one point uniformly to all directions with gradual attenuation.

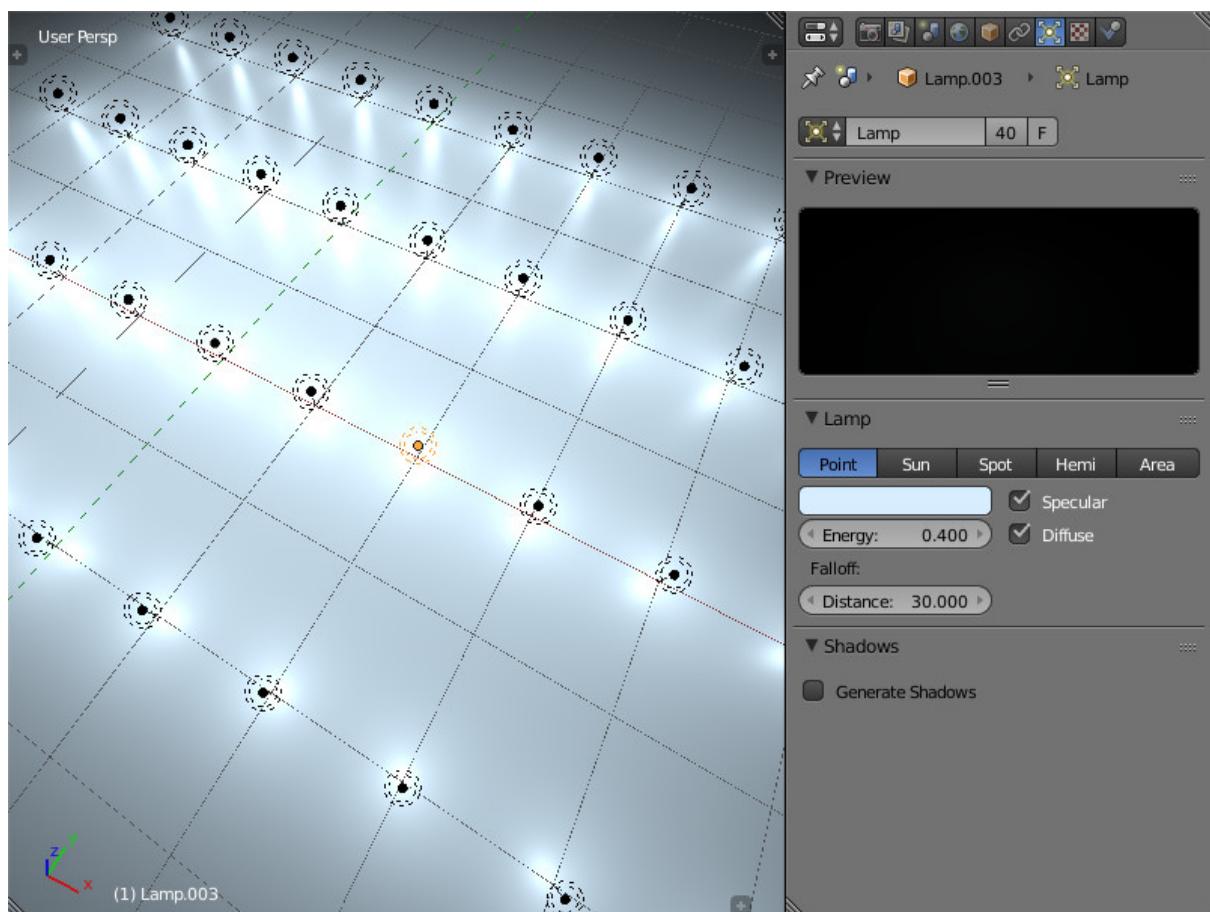
Sun Light propagates from an infinite plane in one direction without attenuation.

Spot Light propagates from one point within the angular limit, with gradual attenuation.

Hemi Hemispherical. Light propagates from an infinite hemisphere without attenuation.

Light source setup

Performed in the Object Data tab when a lamp object is selected.



Color Light color. The default value is (1.0, 1.0, 1.0) (i.e. white).

Energy Radiation intensity. The default value is 1.0.

Falloff Attenuation type. The value is exported but the engine always uses Inverse Square. It is applicable to the Point and Spot light source types. The default value is Inverse Square.

Distance Attenuation parameter. It is applicable to the Point and Spot light source types. The default value is 25.0.

Specular Create specular highlights. Enabled by default.

Diffuse Do diffuse shading. Enabled by default.

Spot Shape > Size Cone angle in degrees. It is applicable to the Spot light source type. The default value is 45°.

Spot Shape > Blend Parameter for blurring light spot edges. It is applicable to the Spot light source type. The default value is 0.15.

Dynamic Intensity Use this light source for calculating the time of day. Applicable only to the Sun light source type. Disabled by default.

Shadow > Shadow Use this light source for shadow calculation. Should be used when multiple light sources are present. Disabled by default.

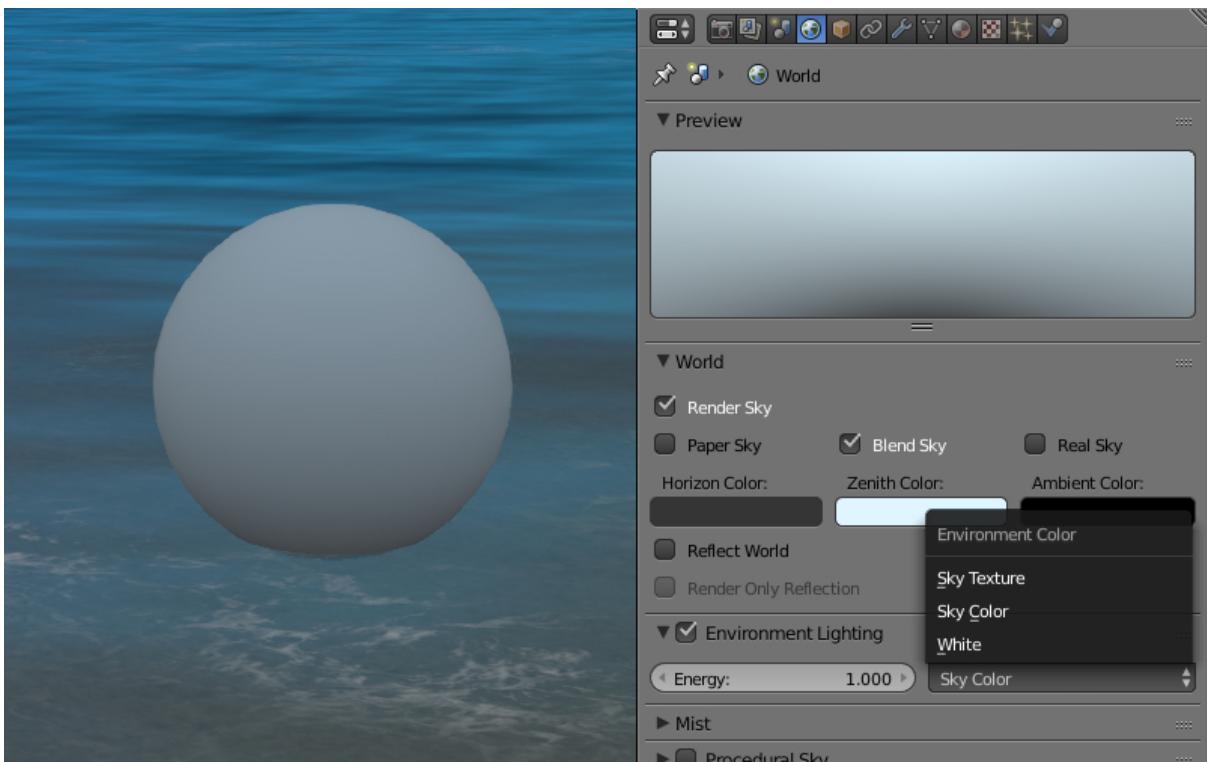
Environment Lighting (Ambient)

The engine supports 3 methods of the environment lighting simulation.

1. Flat white lighting.
2. Hemispherical lighting model in which horizon and zenith colors should be specified. As a result objects are filled with a gradient between these two colors depending on the direction of normals.
3. Lighting using an [environment map](#) - so called image-based lighting.

Activation

Enable the Environment Lighting checkbox on the World tab.



Setup

World > Environment Lighting > Energy Environment lighting intensity. The default value is 1.0.

World > Environment Lighting > Environment Color Selection of the environment lighting simulation method: White - flat white lighting, Sky Color - hemispherical model, Sky Texture - lighting using an [environment map](#). The default value is White.

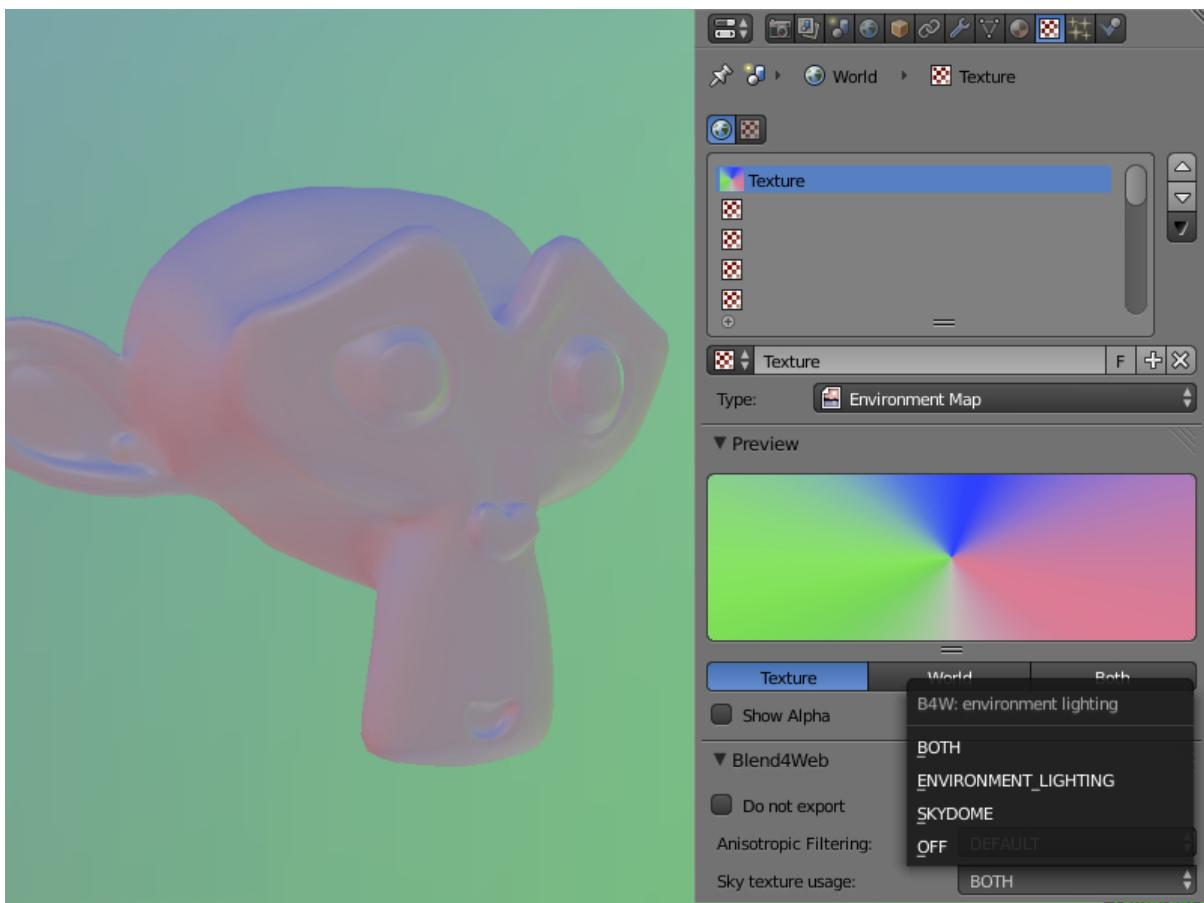
World > Horizon Color and World > Zenith Color If the hemispherical model (Sky Color) is selected the horizon and zenith colors can be specified by means of the World > Horizon Color and World > Zenith Color color pickers. It is recommended to activate the World > Blend Sky option for better color selection.

Environment map method

To use an environment map for environment lighting:

1. Enable the Environment Lighting checkbox on the World tab.
2. Select Environment Lighting > Sky Texture.
3. Go from the World tab to the Texture tab.
4. Create an [environment map](#), load the corresponding image to it.
5. For the environment map select ENVIRONMENT_LIGHTING or BOTH as the Sky Texture Usage value on the Export Options panel (the BOTH option also

enables using this texture as a skydome texture).

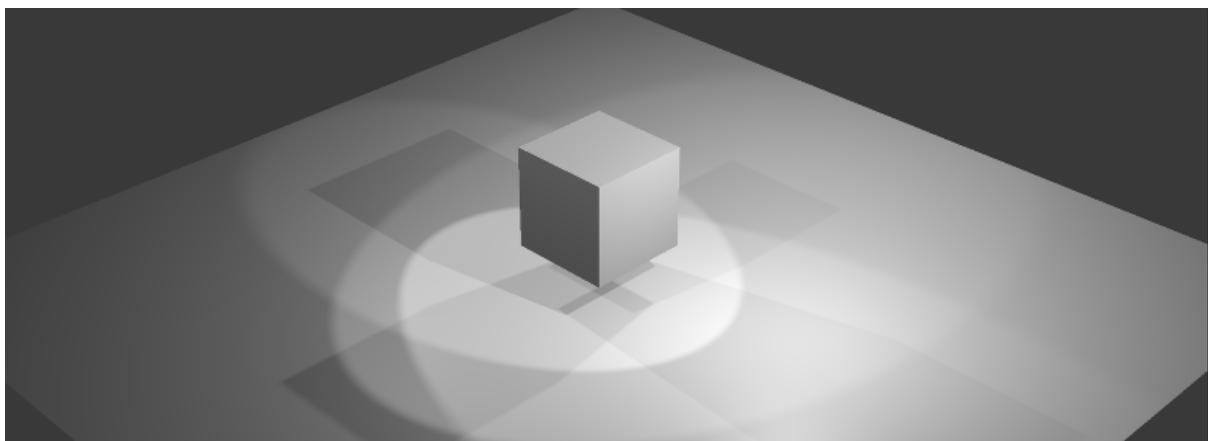


Shadows

Shadows are exceptionally important for rendering the final picture. They provide the viewer not only with information about the objects' outline but also about their height and relative position, light source position and so on.

Blend4Web implements the following shadow rendering techniques: cascaded shadow maps (CSM) and softened shadows (PCF).

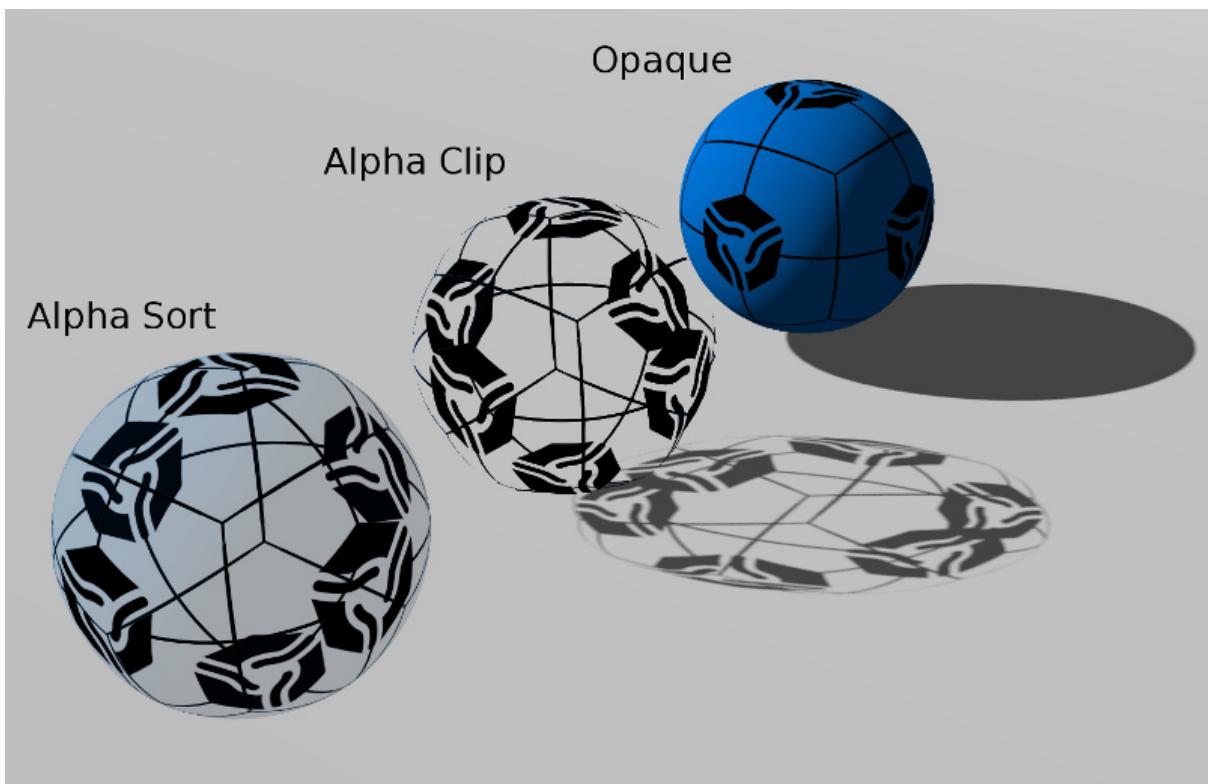
Up to 4 (or 3 if SSAO is enabled) light sources can generate shadows simultaneously. If Shadow parameter is enabled for more than 4 light sources, shadows will still be generated only from 4 of them.



Activation

1. Enable the Shadows: Cast checkbox under the Object tab for the objects which cast shadows.
2. Enable the Shadows: Receive checkbox under the Object tab for the objects which receive shadows.
3. Make sure that the Shadows option in the Render tab has the value AUTO or ON.

Note: Objects, which have [transparent materials with a gradient](#), do not cast shadows.

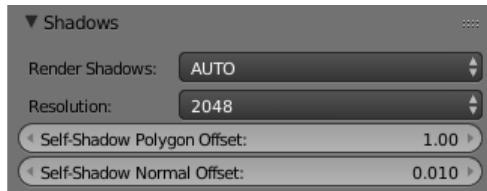


Setup

Direction If there are multiple light sources, it is recommended to specify the exact light source which is used for shadow calculations, by enabling the **Shadow > Shadow** checkbox under the Object Data tab for the selected lamp object.

Color The shadow color is determined by the [environment lighting](#) settings.

The following additional settings are located on the Shadows panel of the Render tab:



Resolution Shadow map resolution. The default value is 2048x2048px.

Self-Shadow Polygon Offset Coefficient for shifting polygons relative to light source orientation. The default value is 1.

Self-Shadow Normal Offset Coefficient for shifting polygons along their normals. The default value is 0.010.

The last two parameters can be used to reduce self-shadowing artifacts. These artifacts appear for the objects that cast and receive shadows at the same time. The Self-Shadow Polygon Offset parameter is more effective for fighting against artifacts in inner areas of polygons while Self-Shadow Normal Offset is better for the boundary areas. Both these parameters lead to shadow distortions so we recommend to set them as low as possible.



Note: Shadows from Point light sources are generated the same way as from Spot light sources and are projected only in one direction specified by the source's Rotation parameter.

Enable CSM Activates the using of cascaded shadows model; reveals additional options.

Disabled by default. Won't work if the Shadow setting is enabled for more than one light source. Point and Spot type light sources support only one shadow cascade.

This option allows to choose between the following shadow generation models:

- Generic model which uses a single optimized shadow map for the whole scene (Enable CSM is turned off).
- Shadow cascades (Enable CSM is turned on).

Blur Radius Blur ratio for setting up softened shadows. The default value is 3. Zero value produces hard shadows.



Softened shadows can improve visual quality and realism. They hide the jugged edges inevitable when using image-based techniques, that is especially noticeable for low-resolution shadow maps. The using of softened shadows often allows to decrease resolution without substantial quality loss.

Generic shadows

This option suits well smaller scenes with a limited number of objects. Thanks to optimizations applied for such scenes, one can achieve better shadow quality as compared with cascaded shadows. Also, this option is simpler and faster for setting up, while using a single shadow map greatly improves the performance.

Shadow cascades

Note: These settings are supported only for Sun light sources. Cascades are turned off for other types of light sources.

In order to provide acceptable shadow quality and to cover considerable space at the same time it is required to use multiple stages for shadow generation (cascades). Thus, the best quality cascades are situated near the observer while the worst quality cascades are in the distance. This option suits well middle-to-large scenes, e.g. game levels.

When enabled the following extended settings are revealed:



CSM Number Number of shadow cascades. From 1 to 4 cascades are supported. The default value is 1.

CSM First Cascade Border First cascade size. The default value is 10.0.

CSM Last Cascade Border Last cascade size. The default value is 100.0.

The sizes of the intermediate cascades are interpolated between the two above-mentioned parameters.

Note: When setting up the shadows keep in mind that the bigger the cascade size is, the worse and less detailed are the shadows inside it. On the other hand, reducing the CSM First Cascade Border parameter makes the subsequent less detailed cascades closer to the camera and thus more noticeable. Reducing the CSM Last Cascade Border parameter forces shadows to disappear at more close distance from the camera. However, when softened shadows are used the overall quality will improve thanks to blurring at the edges.



CSM first cascade border: 7



CSM first cascade border: 20

CSM First Cascade Blur Radius Blur ratio for the first cascade. The default value is 3. Zero value produces hard shadows.

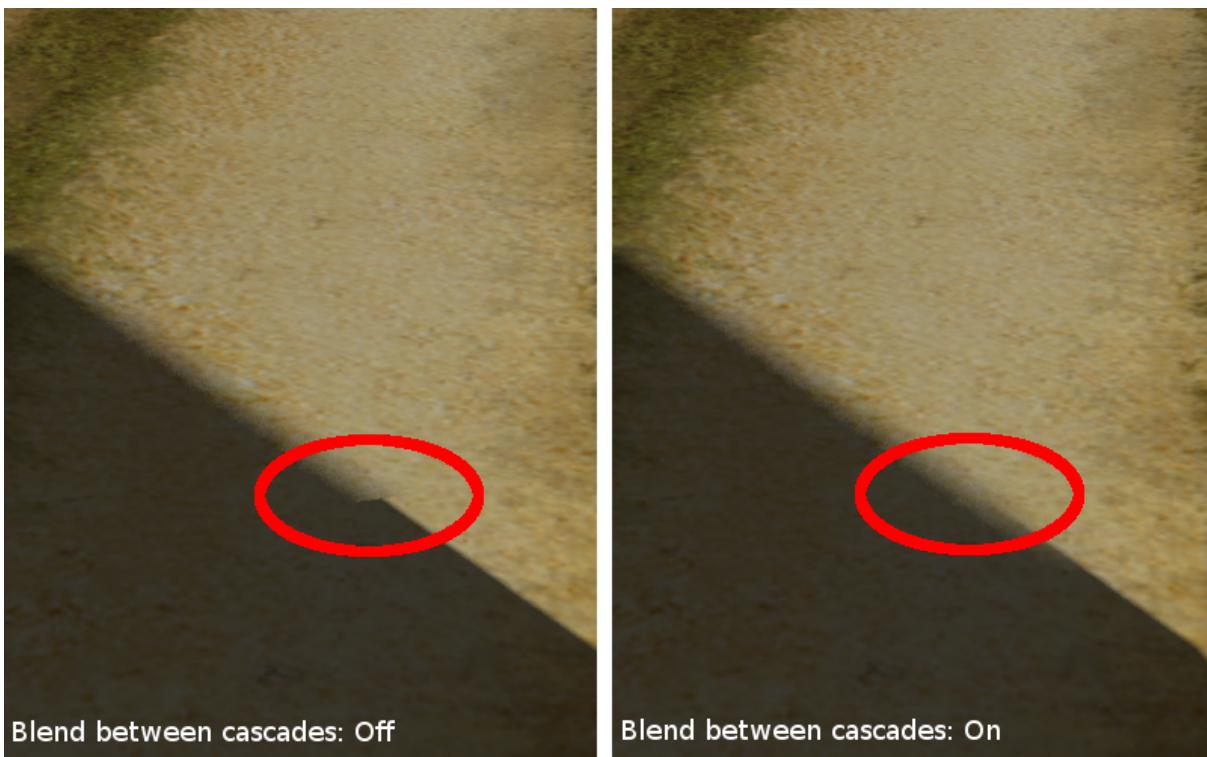
CSM Last Cascade Blur Radius Blur ratio for the last cascade. The default value is 1.5. Zero value produces hard shadows.

The blur radii of the intermediate cascades are interpolated between the two above-mentioned parameters.

Note: We recommend to start setting up the softened shadows with the first cascade (using CSM First Cascade Blur Radius) and then proceed to other cascades (using CSM Last Cascade Blur Radius). Often the last cascade may need less blurring than the first one. This may be needed to prevent the shadows on the last cascade being too faded due to low resolution. This also reduces undesirable self-shadowing artifacts.

Fade-out Last Cascade Smooth dying-out of the last cascade. Enabled by default.

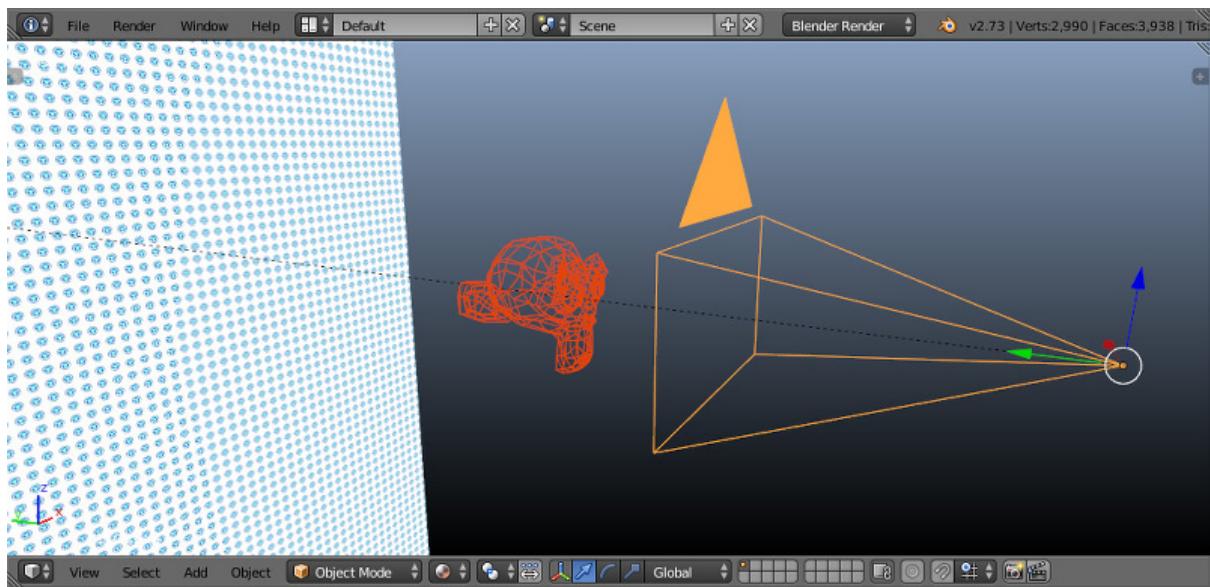
Blend Between Cascades Smoothing the boundaries between the cascades. Enabled by default.



Background

You can change the background in the following ways:

1. Enable World > Render Sky, then set the Horizon Color and the Zenith Color under Blender's World tab.
2. Place the whole scene inside a model (e.g. a cube or a sphere) with its normals directed inside, with a material and an optional texture.
3. Place a surface with a material and an optional texture in front of the camera. Parent it to the camera. If required, tweak the distance to this surface, starting and ending clipping planes for the camera.



4. Use a [skydome](#).
5. Set up the procedurally generated [atmosphere](#).
6. Set the engine's `background_color` parameter with the `config.set()` method. Please note, that World > Render Sky under Blender's World tab must be disabled. This value is used as argument for the WebGL `clearColor()` method. For correct results, it's recommended to turn the WebGL context transparency off (the `alpha` parameter). Such the configuration is used by default in the engine's standard web player.

```
var m_cfg = b4w.require("config");
var m_main = b4w.require("main");

// gray
m_cfg.set("background_color", new Float32Array([0.224, 0.224, 0.224, 1.0]));
m_cfg.set("alpha", false);

m_main.init(...);
```

7. You can use any HTML content behind the canvas element, to which the rendering is performed, as a background. To do this, activate the WebGL context transparency (the `alpha` parameter). Please note, that World > Render Sky under Blender's World tab must be disabled. For correct results, it's recommended to set absolutely transparent black background color. Such the configuration is used by default in the [scene viewer](#) of Blend4Web SDK.

```
var m_cfg = b4w.require("config");
var m_main = b4w.require("main");

m_cfg.set("background_color", new Float32Array([0.0, 0.0, 0.0, 0.0]));
m_cfg.set("alpha", true);

m_main.init(...);
```

See also:

[Alpha Compositing](#)

Postprocessing Effects

Table of Contents

- Postprocessing Effects
 - Motion Blur
 - Depth of Field
 - Screen-Space Ambient Occlusion
 - God Rays
 - Bloom
 - Outlining
 - Glow
 - Anti-Aliasing

Motion Blur

The motion blur effect can be used to improve the realism of an interactive scene. It is displayed as picture blurring when the camera or objects move.



Activation

Activate the Motion Blur panel on the Render tab.

Additional settings

Factor Effect appearance ratio. The higher this value is the stronger is the motion blur.

Decay Threshold Blur fade-out ratio. The higher this value is the more distinct is the effect. The default value is 0.01.

Depth of Field

The depth of field effect (DOF) can be used to accentuate a part of a scene. It is displayed as picture blurring nearer and further from the camera focus.



Activation

1. Select an active camera and go to its settings panel Camera (Object Data).
2. Then two options are available:
 - Select an object to use as the camera's focus in the Focus option of the Depth of Field panel. In this case moving away or approaching this object will cause a corresponding correction of the camera focus.
 - Set a non-zero value for the Distance on the same panel (in Blender units = meters). In this case the camera focus will be located at this distance from the camera and will move together with it.

Additional settings

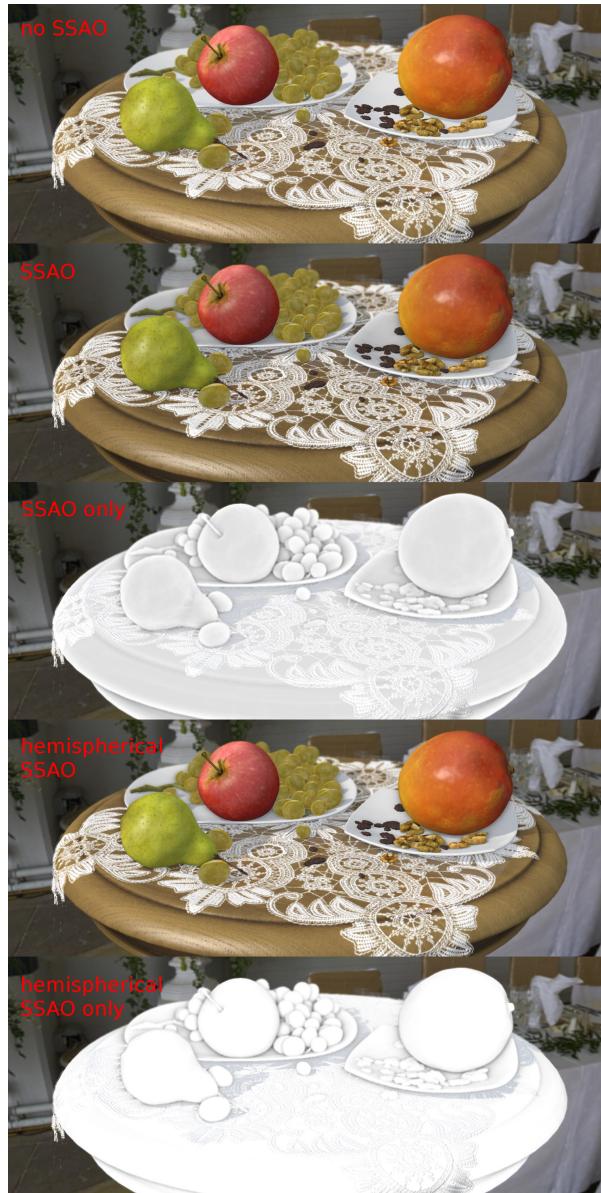
Front The distance from the focus to the nearest plane (relative to the camera) behind which full blurring occurs (in meters). The default value is 1.0.

Rear The distance from the focus to the furthest plane (relative to the camera) behind which full blurring occurs (in meters). The default value is 1.0.

Power Blurring ratio. The default value is 3.0.

Screen-Space Ambient Occlusion

The screen-space ambient occlusion (SSAO) effect can be used to fake complex light reflections from objects. The basis of this effect is that the space between close objects is less accessible for diffused light and hence is darker.



Activation

Activate the Ambient Occlusion SSAO panel under the Render tab and set the Render Shadows parameter to AUTO or ON on the Rander > Shadows panel.

Additional settings

Radius Increase The spherical sampling radius multiply factor when transferring from the internal sampling ring to the external one. The default value is 3.0.

Use Hemisphere Use a hemispherical sampling for shading instead of a spherical. Besides, it uses different shading law.

Use Blur Depth Test Use edge-preserving blur to SSAO if flag will be turned on. Otherwise, it uses blur which averages a 4x4 rectangle around each pixel.

Blur Depth Test Discard Value Influence of depth difference between samples on blur weight. It uses with Use Blur Depth Test activated flag. The default value is 1.0.

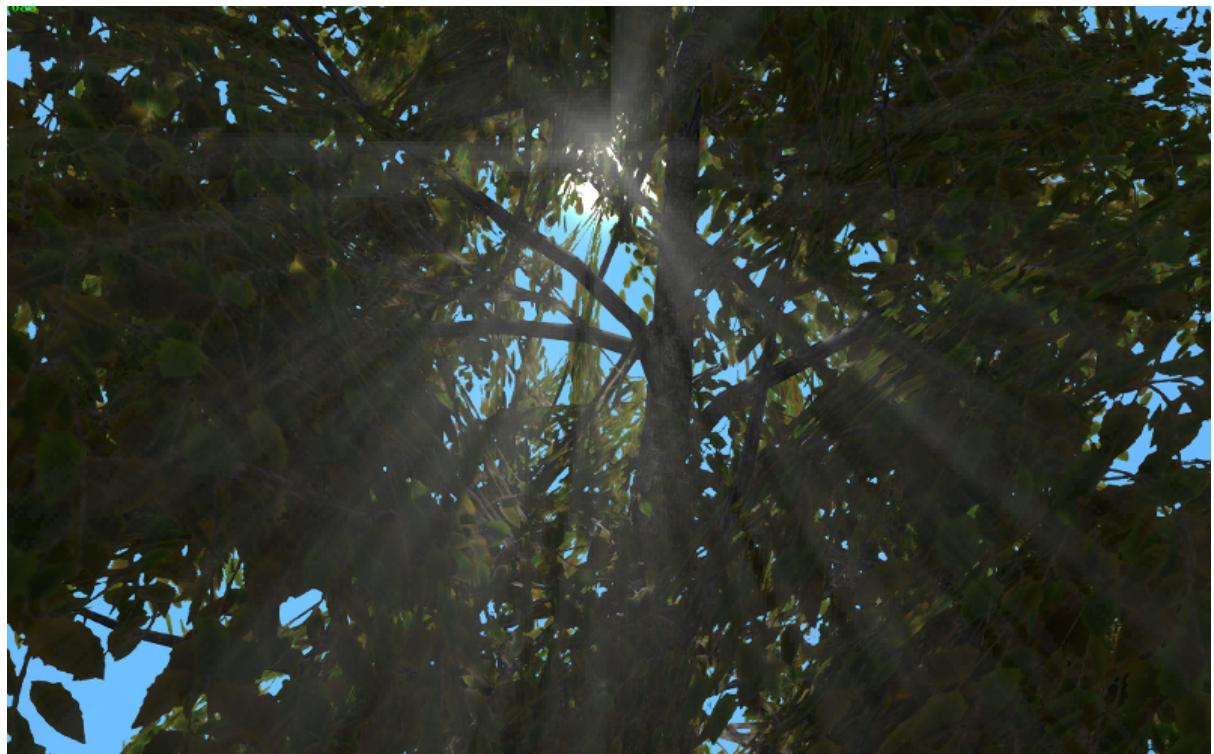
Influence SSAO appearance factor. The default value is 0.7.

Distance Factor Factor of SSAO decay with distance. The default value is 0.0 (i.e. no decay).

Samples Number of samples (the more samples there are the better is the quality but the poorer is the performance). The default value is 16.

God Rays

The god rays effect (aka crepuscular rays) simulates well-known natural phenomenon - the shining of illuminated air parts.



Activation

Activate the God Rays panel under the Render tab.

Additional settings

Intensity The effect appearance factor. The default value is 0.7.

Maximum Ray Length Rays length factor. Defines the step between samples of radial blurring. The default value is 1.0.

Steps Per Pass Number of steps per single sample. The default value is 10.0.

Bloom

Bloom appears when a picture has elements with a very different brightness. A glowing halo is created around the bright details.



Activation

Activate the Bloom panel under the Render tab.

Additional settings

Key Bloom intensity.

Blur Bloom blurriness factor.

Edge Luminance The boundary value of an element's relative brightness above which the bloom effect appears.

Outlining

As a result of the outline glow effect, a luminous colored halo will be displayed around the object.



Activation

The outlining is activated programmatically via API. Different animation models can be applied such as constant glow, fading out glow, pulsatory glow and any other. In order to enable the outlining effect on a certain object, make sure that the Render > Object Selection panel's Enable property is set to ON or AUTO.

Additional settings

On the Object > Selection and Outlining panel:

Enable Outlining Permit using the outline glow effect on this object.

Duration Duration of glow animation, seconds. The default value is 1.

Period Repeat period of glow animation, seconds. The default value is 1.

Relapses The number of iterations of glow animation. If zero, animation is repeated forever. The default value is 0.

Outline on Select Activate glow animation upon selecting the object. In this case the Selectable option must be enabled. In case of a user-defined glow animation model, this option must be disabled in order to avoid conflict.

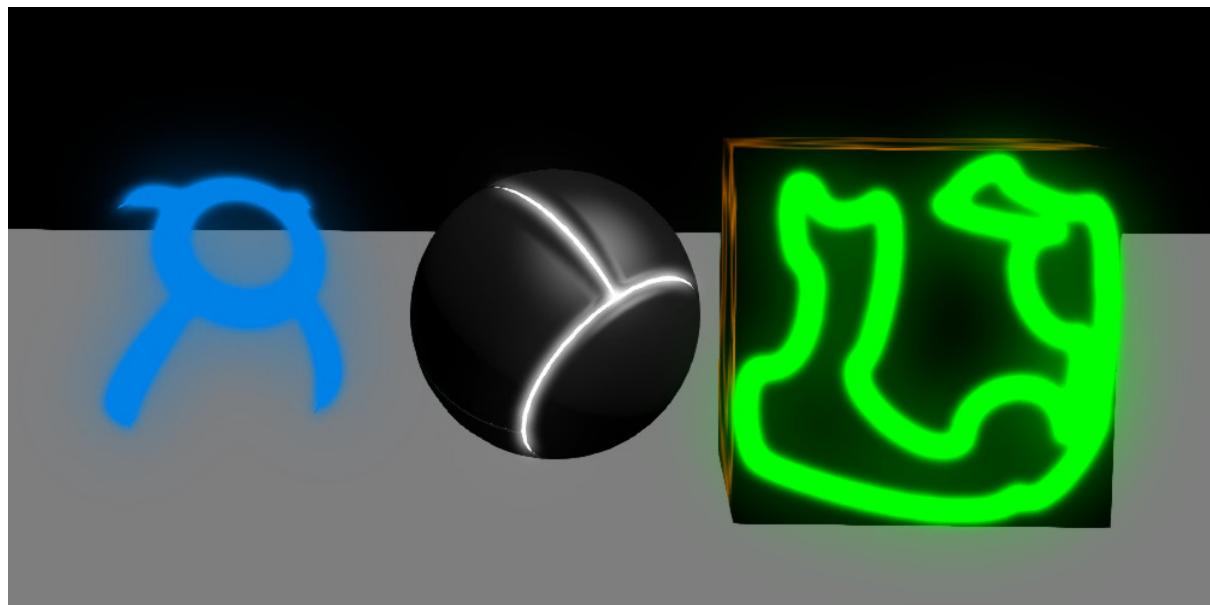
On the Render > Object Outlining panel:

Factor When this parameter decreases so does the thickness and the brightness of the halo around the object. The default value is 1.

The Render > Object Outlining settings are taken as default when the glow effect is initiated via API.

Glow

Effect of halo which is observed around emissive objects due to light scattering in the atmosphere and inside the human eye.



Activation

Add a `B4W_GLOW_OUTPUT` node to a node material. The Enable Glow Materials option on the Render > Glow Materials panel should be set to ON or AUTO.

Additional settings

Small Mask: Intensity Intensity of glow obtained through the smaller mask. The default value is 2.0.

Small Mask: Width Width of glow obtained through the smaller mask. The default value is 2.0.

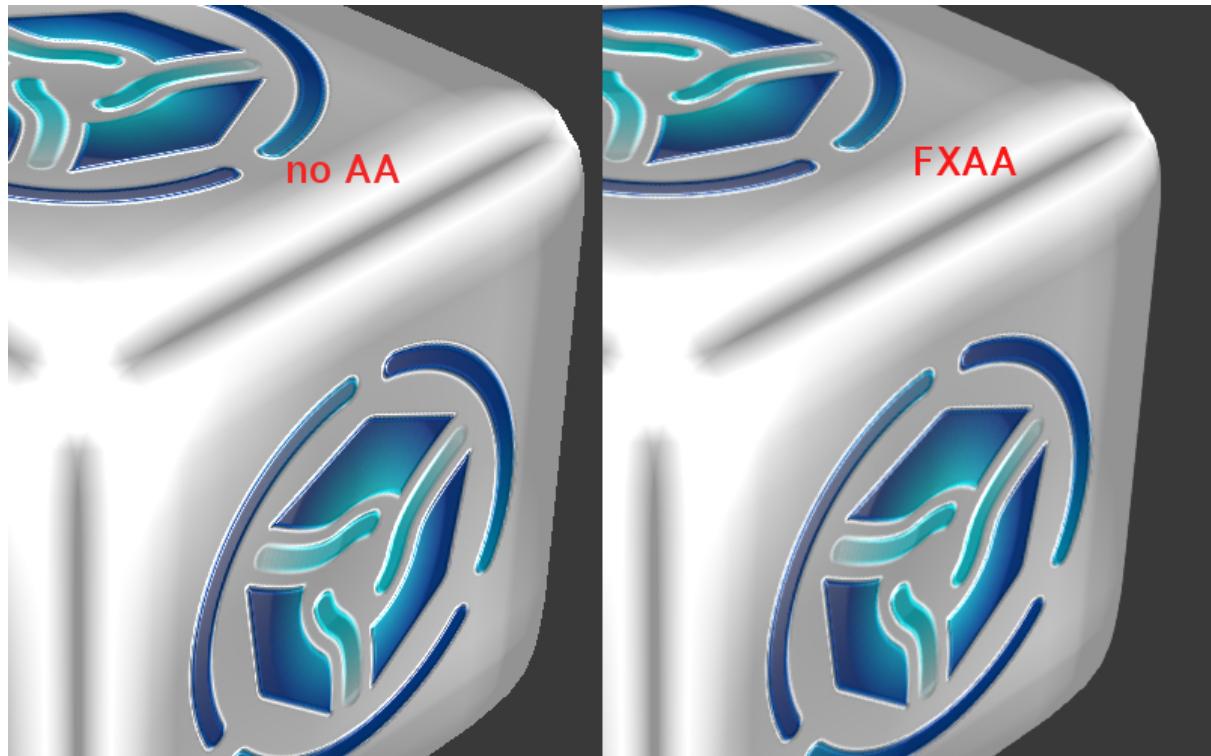
Large Mask: Intensity Intensity of glow obtained through the larger mask. The default value is 2.0.

Large Mask: Width Width of glow obtained through the larger mask. The default value is 6.0.

Render Glow Over Transparent Objects Render the glow effect over transparent objects.

Anti-Aliasing

Anti-aliasing is used to reduce undesirable rendering artifacts (poor pixelization).



Activation

Select quality profile using AA Quality menu located on the Render > Anti-Aliasing panel.

- None - disable anti-aliasing,
- Low, Medium, High - enable anti-aliasing with the given quality profile.

Medium profile is used by default.

Additional settings

The anti-aliasing method is assigned simultaneously with the selection of the engine performance profile:

- low quality - no anti-aliasing,
- high quality and ultra quality - use the FXAA 3.11 algorithm (Fast Approximate Anti-Aliasing by Nvidia).

Stereo Rendering

Table of Contents

- Stereo Rendering
 - Activation
 - Additional Settings

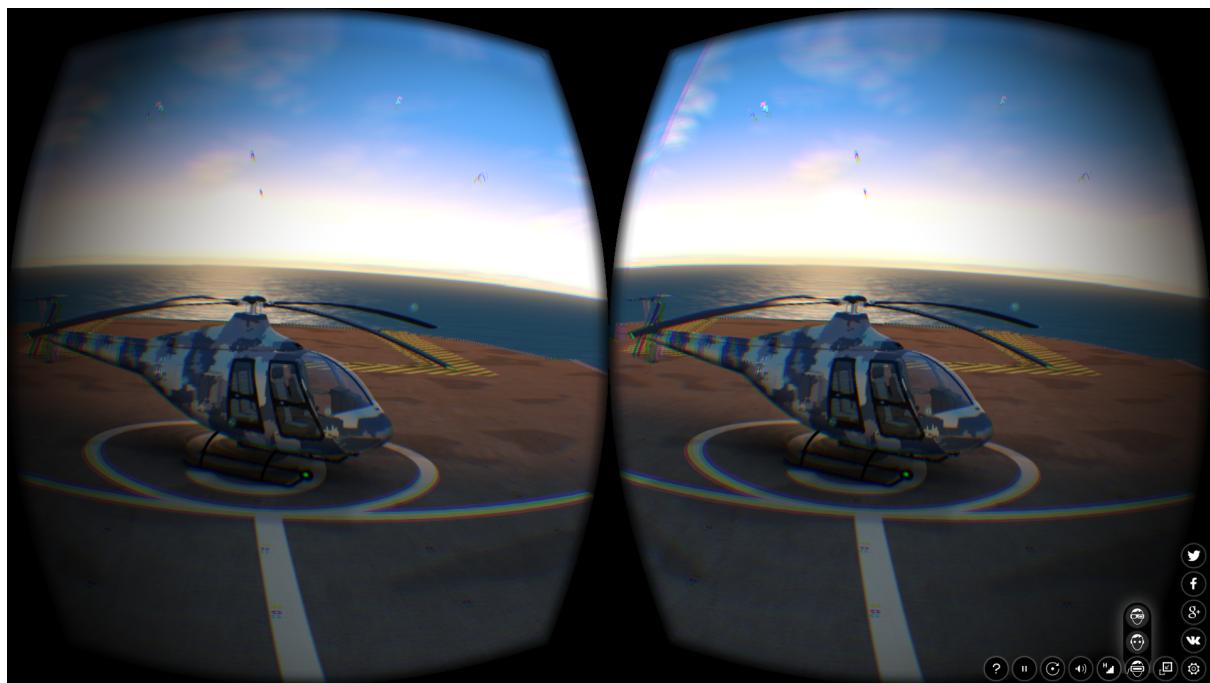
The stereoscopic rendering mode is intended for viewing the content using special glasses. It is activated by an application via API.

Blend4Web supports two techniques of the stereo image rendering - anaglyph image and the HMD (head-mounted display).

Anaglyph:



HMD:



HMD is an experimental feature, for now it works only with the Eye type cameras.

Activation

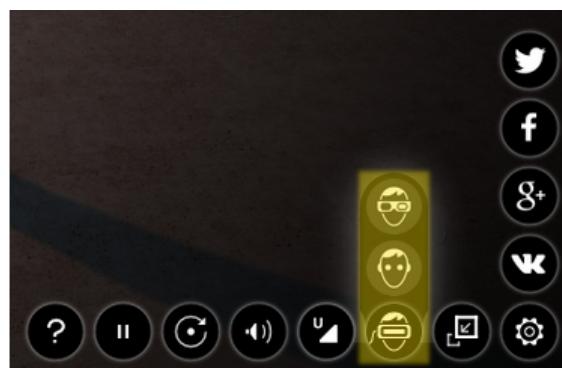
To use HMD stereo rendering, you need to install Oculus's [runtime utility](#). Windows and MacOS versions can be found on the Oculus website in binary format, while Linux version should be compiled from the source code.

For now, the HMD is supported by the [Chromium experimental builds](#) and in the [Firefox nightly builds](#).

Chromium settings.

Firefox settings.

To turn the stereo rendering on, you need to choose certain option in the settings, in the third column from the right, as shown on the picture.



For stereo rendering to work correctly, switching to the full screen mode is recommended.

Additional Settings

None.

CHAPTER 20

Textures

Table of Contents

- [Textures](#)
 - [Texture Types](#)
 - [Generic Settings](#)
 - [Diffuse Map](#)
 - * [Activation](#)
 - * [Additional Settings](#)
 - [Specular Map](#)
 - * [Activation](#)
 - * [Additional Settings](#)
 - [Normal Map](#)
 - * [Activation](#)
 - * [Additional Settings](#)
 - [Height Map. Parallax Mapping](#)
 - * [Activation](#)
 - * [Additional Settings](#)
 - [Stencil Map](#)
 - * [Activation](#)
 - * [Additional Settings](#)
 - * [Limitations](#)
 - * [Example](#)
 - [Video Textures](#)
 - * [Supported formats \(containers\):](#)
 - * [Setting up the Texture](#)
 - * [Specifics of Mobile Devices](#)
 - [Environment Map](#)
 - * [Making Environment Maps](#)
 - [Mirror Map](#)
 - * [Activation](#)
 - * [Additional Settings](#)
 - [Skydome](#)
 - * [Activation](#)
 - * [Additional Settings](#)
 - [Special Texture Types](#)
 - * [3D scene](#)
 - * [Canvas](#)

Textures are hand-made or procedurally generated images that can be applied to the model surfaces to add more detail. As a rule, the image pixels are assigned to the 3D surface points using texture mapping. For this reason they are sometimes referred to as maps.

Usually the textures are placed into material texture slots. They can be also used for particle systems parametrization and for creating the skydome.

Texture Types

The Type drop-down menu (for selecting texture type) is located under the Textures tab. The engine supports the following texture types:

1. Image or Movie

In this case, the texture is defined by an image or a video file. Following file formats are supported:

- .PNG
- .JPG

It can be used for the following purposes:

- diffuse map
- specular map, this can also be packed into the alpha channel of a diffuse texture
- normal map
- height map; this must be packed into the alpha channel of a normal map; it is used for visualization of relief surfaces ([parallax mapping](#)).
- stencil map
- video texture

2. Environment Map

- mirror map
- skydome texture
- used for implementation of an environment lighting method

3. None

- applied to the Blender's default scene cube. It is also used for rendering [a scene to texture](#) and for [rendering canvas textures](#).

4. Blend, gradient

- is used in [particle systems](#)

Generic Settings

Dimensions Bitmap dimensions for image textures (image width and height in pixels) should be a 2^N number, i.e. 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096 px. Using textures with other dimensions (so-called NPOT) is supported but is not recommended. Dimensions should be at least 4 pixels for the correct texture compression. Normally square images are used (e.g. 512 x 512 px), however rectangular

ones can be used too (e.g. 4 x 128 px). Using images bigger than 2048 px is not recommended.

Image Mapping > Extension Texture coordinates interpretation mode (Wrap Mode in WebGL). This is available for Image or Movie texture type. In case of Repeat value the engine sets the REPEAT mode for the texture. In this case the integer part of the texture coordinates is ignored and the fractional part is used. In all other cases (for example Extend) the engine sets the CLAMP_TO_EDGE mode. In this case the texture coordinates are limited by the [0, 1] segment. The default value is Repeat.

Mapping > Coordinates Texture coordinates type. Supported types are UV (use UV map), Normal (use direction at the camera; available only for diffuse maps; used for the creation of material capture, matcap) and Generated. The default value is Generated.

Mapping > Size Scaling the UV map along respective axes. The default values are 1.0.

Export Options > Do Not Export Do not export the texture.

Export Options > Disable Compression Disable texture compression for this texture. Used in cases when [texture compression](#) deteriorates the image quality. For example it's recommended to disable compression for mask textures used to mix different parts of materials.

Export Options > Shore Distance Map Used in [outdoor rendering](#).

Export Options > Anisotropic Filtering Anisotropic filtering factor for the individual texture. It has priority over the similar parameter for the scene. The default value is DEFAULT (i.e. use the scene settings).

Water Foam The foam texture. Used by the water rendering material.

Note: Texture compression is disabled for textures used as normal maps.

Diffuse Map

A diffuse map is used for specifying scattered light distribution (the Lambert model).

Activation

Enable the Diffuse > Color checkbox on the Textures > Influence panel.

Additional Settings

Influence > Diffuse > Color Influence of the texture on the diffuse color. The default value is 1.0.

Influence > Blend The type of the interaction with the material color (Material > Diffuse > Color), or with the vertex color if the Vertex Color Paint checkbox is enabled. The following types are supported: Mix (mixes with the color), Multiply (multiplies by the color). The default value is Mix.

Specular Map

The specular map is used for specifying the reflected light color distribution (the Phong model).

Activation

Enable the Specular > Color checkbox on the Textures > Influence panel.

Additional Settings

Influence > Specular > Color The influence of the texture on the reflected light color. The default value is 1.0.

Influence > Blend The type of interaction with the reflected light color of the material (Material > Specular > Color). Mix (mixes with the color) is the only supported type. The default value is Mix.

The specular map can be packed to the alpha channel of a diffuse texture for optimization purposes. In such case it is required for the texture to enable the Diffuse > Color and Specular > Color checkboxes simultaneously. The color range is limited by gray tints.

Normal Map

A normal map is used for specifying the distribution of surface normals (perpendiculars) with the purpose of the relief detailization. The information about the normals should be stored in the texture space of coordinates. Normal maps baked in the object space of coordinates are not supported.

Activation

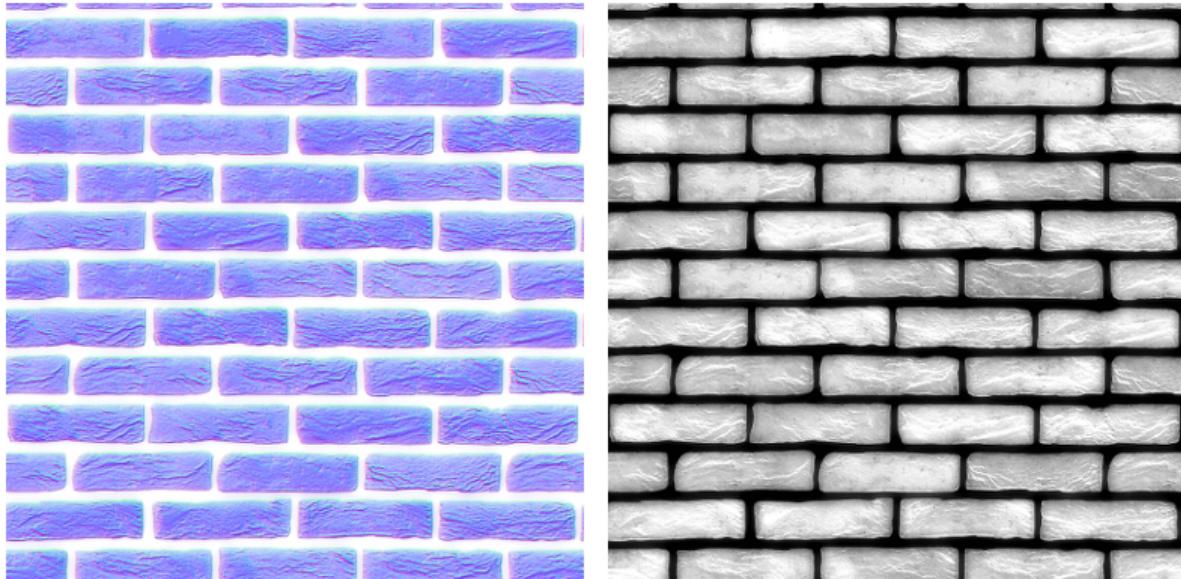
Enable the Geometry > Normal checkbox on the Textures > Influence panel.

Additional Settings

Influence > Geometry > Normal Normal map influence on the resulting normals calculation. The default value is 1.0.

Height Map. Parallax Mapping

A height map contains information about the distribution of relative relief heights. The higher the surface level is, the brighter is its color. A height map combined with a normal map is required for the implementation of relief surface effect (parallax mapping). A height map should be present in the alpha channel of a normal map.



Activation

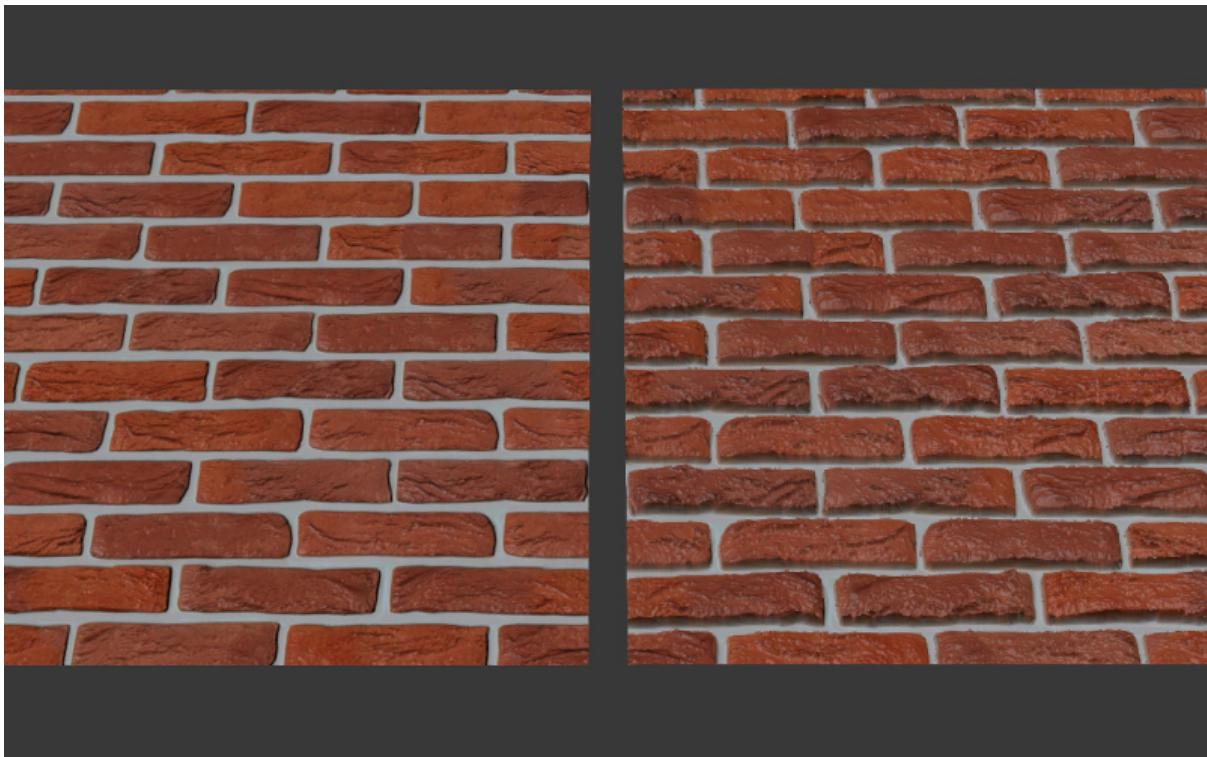
For the normal map enable the Parallax panel in addition to the Geometry > Normal checkbox.

Additional Settings

Parallax > Parallax Scale Influence factor for the relief surface effect. The default value is 0.03.

Parallax > Parallax Steps The number of iterations for the relief surface calculations. Bigger value leads to better quality but is more computationally expensive.

Parallax > Parallax LOD distance Distance at which the parallax effect is observed.



Stencil Map

The special purpose texture (colorful or grayscale) contains information about the distribution of other texture surfaces.

Activation

1. In case of node materials a stencil map should be used in the corresponding node structure.
2. In case of generic materials a stencil map should be located in a texture slot between two mixed diffuse textures. A stencil map requires to set both the RGB to Intensity and the Stencil checkboxes on the Textures > Influence panel.

Additional Settings

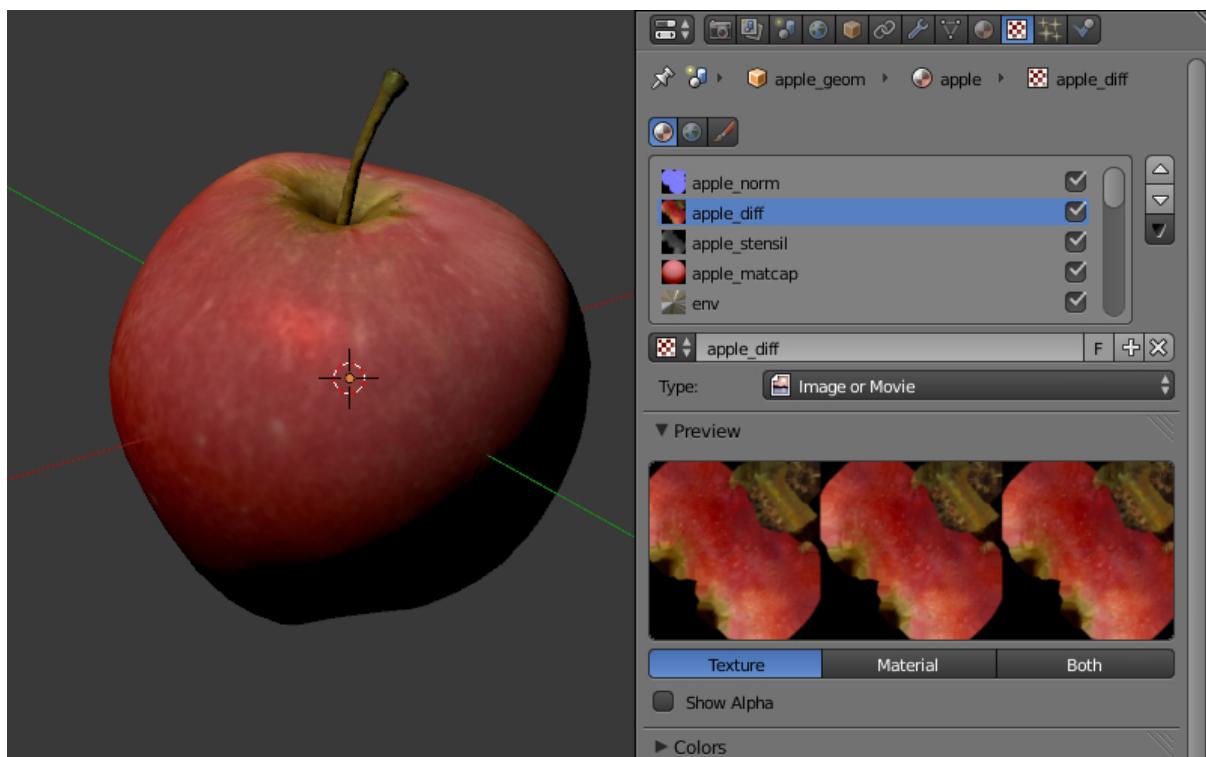
In the case of generic materials one of the mixed diffuse textures can have the Normal (“matcap”) texture coordinates type.

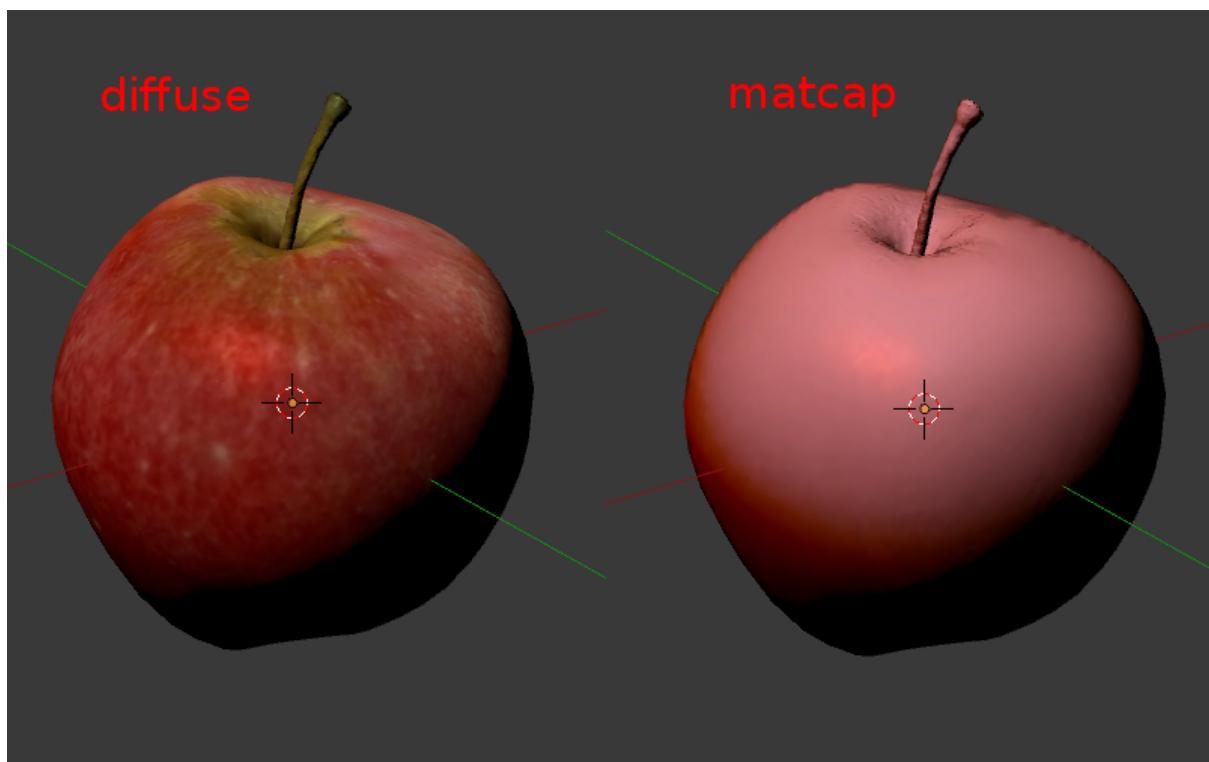
Limitations

In case of generic materials the engine only interprets the red channel of a stencil map. Specular maps or normal maps (if any) are not being mixed. The Mapping > Size setting is extracted from the first texture and is applied to all remaining textures.

Example

The apple model material has the following textures: a normal map, a diffuse texture with a specular map in its alpha channel, a stencil map, a diffuse “matcap” map, an environment map.





Video Textures

A video file can be used as a texture if Image or Movie texture type is selected.

Note: Video textures support playing back just video tracks. Audio tracks should be played back by using a SPEAKER object.

Supported formats (containers):

- webm, VP8 codec (Chrome, Firefox)
- m4v, H.264 codec (Chrome, Safari, IE)
- ogv, Theora codec (Chrome, Firefox)

We recommend to use WebM as a basic format. It is an open standard supported by the majority of browsers and offers good picture quality.

Note: Files saved in mp4 and ogg formats have different extensions for audio and video data: .mp4 and .ogg extensions are used for sounds, .m4v and .ogv - for video.

Converting resources between different formats is described in the [corresponding section](#).

Setting up the Texture

The following settings are available for video textures on the **Texture > Image** panel:

Image > Frames Length of the played fragment in frames.

Image > Offset The number of the frame from which the video playback starts.

Image > Cyclic Start video playback afresh each time it finishes.

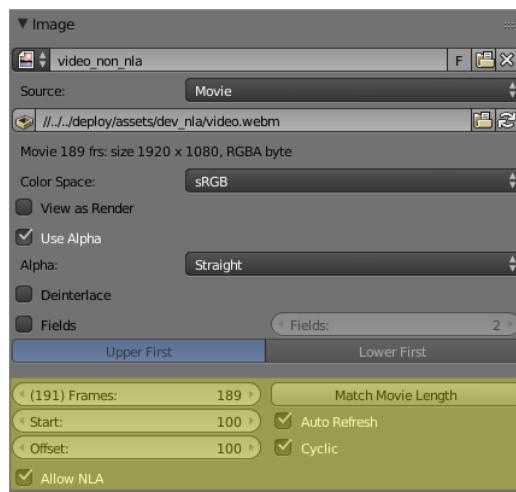
Image > Allow NLA Play back the texture as part of an NLA track. Additionally, enable NLA in the global scene settings through activating **Scene > NLA**. Enabled by default.

For NLA-controlled textures the following option is also applicable:

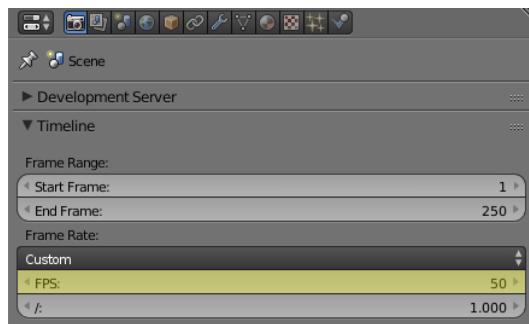
Image > Start Video playback delay (in frames) when using non-linear animation.

For non-NLA-controlled textures the following option is also applicable:

Image > Auto Refresh Play back the video immediately after the scene is loaded.



The video playback rate can be increased. To do this set **Scene > Dimensions > Frame rate** to a value which is different from the FPS value for the videos. Video playback rate is increased proportionally to the ratio of the scene's FPS and the video's FPS.



Note: When video textures are used together with NLA, the video playback can be not corresponding with Blender settings. Namely, there can be observed some lagging withing 5-6 frames due to starting/pausing delay of the <video> HTML element.

Specifics of Mobile Devices

The peculiarities for mobile devices are as follows:

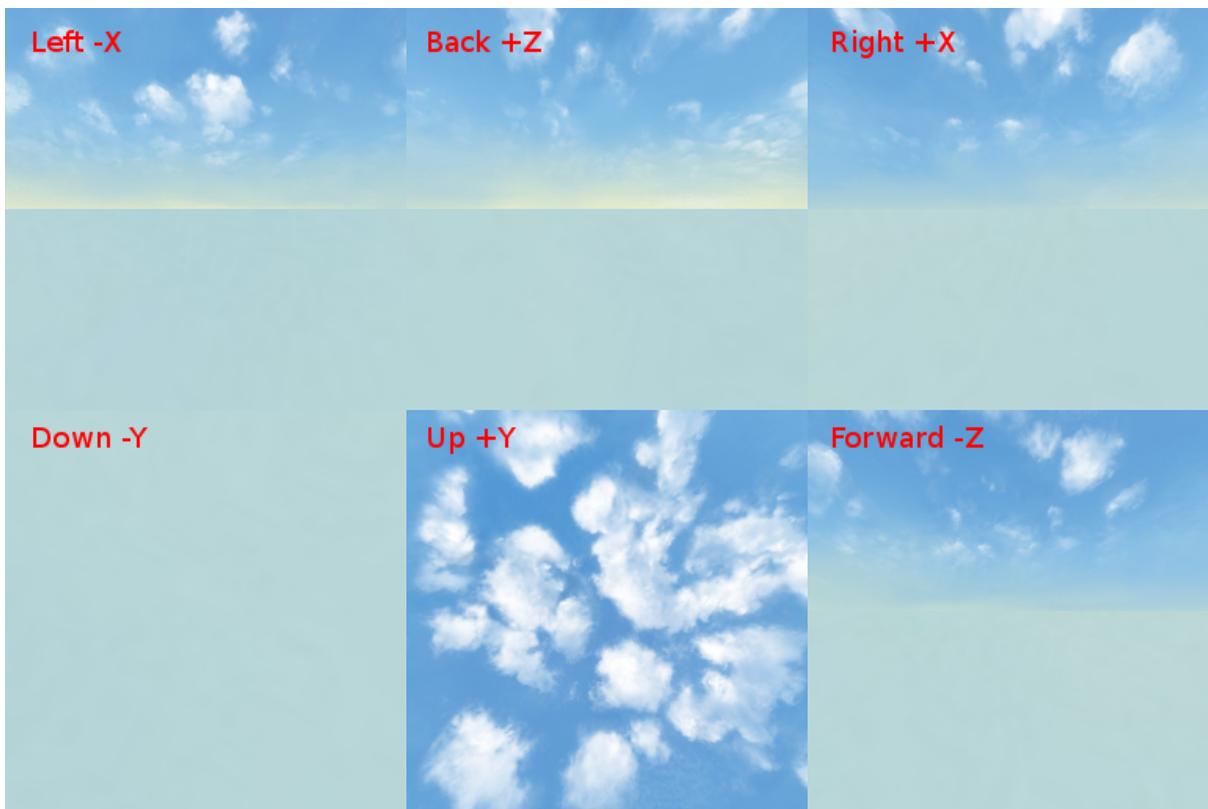
1. Normal operation of video textures on iPhones is not possible because these devices play back videos via the standard iOS video player. For these devices you need to convert your videos to special .seq format by using our [converter](#).
2. some devices only support playing back only one video file.
3. stable operation is not guaranteed if the Offset value is not zero.
4. not all devices support changing the video playback rate.
5. iPad and iPhone do not provide the possibility to control the audio volume for video, and so the audio track should be removed from the video before the file is added to Blender.

Environment Map

An environment map can be used as a [mirror map](#), as a static [sky texture](#) ([skydome](#)) and also for implementation of an [environment lighting](#) method.

The engine considers it as a cube texture. Environment map bitmaps should contain 6 projected environment images, packed in 2 rows 3 pieces in each (a Blender format). Bitmap dimensions for each image should follow the 2^N rule (512, 1024 etc).

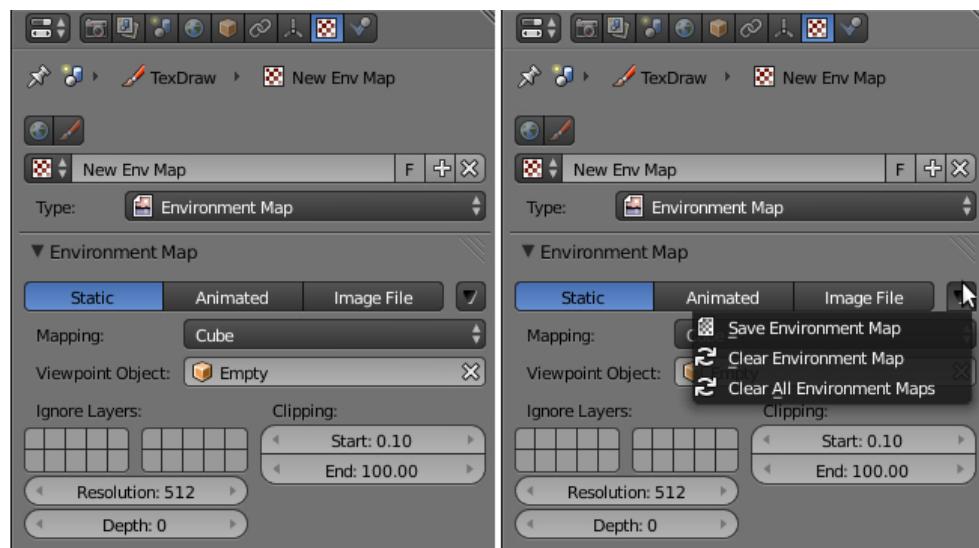
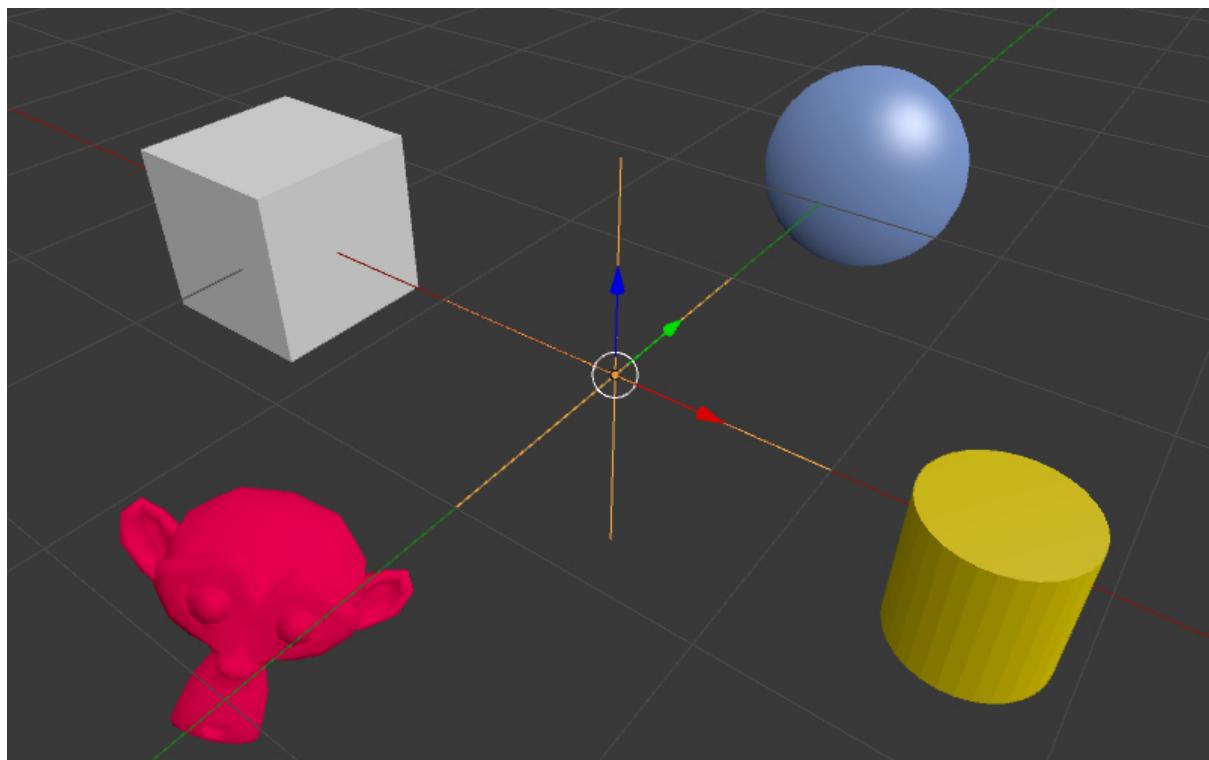
It is recommended to use the lossless format (PNG) in order to avoid seams.



Making Environment Maps

Blender has an option for baking a scene into an environment map. To do this:

1. Create a scene for baking.
2. Add an empty object in the supposed point of view (Add > Empty).
3. Go to the World tab then to the Textures tab and create a new texture with the Environment Map type.
4. On the Environment Map panel select the Static source, then select the empty object in the Viewport Object field, then set the 2^N dimension (512, 1024 etc).
5. Render the scene by pressing F12 (a camera is required).
6. Save the environment map into a file.



Mirror Map

A mirror map is used to visualize the surface reflection. This is an environment map.

Activation

Select the Environment Map texture type (Type). Enable the Shading > Mirror checkbox on the Textures > Influence panel.

Additional Settings

Influence > Shading > Mirror The degree to which the mirror map affects the reflection.
The default value is 1.0.

See also:

[Static reflection](#).

Skydome

A skydome is used to visualize an infinitely far environment (for example the sky). This is an [environment map](#).

Can be also used to implement one of the [environment lighting](#) methods.

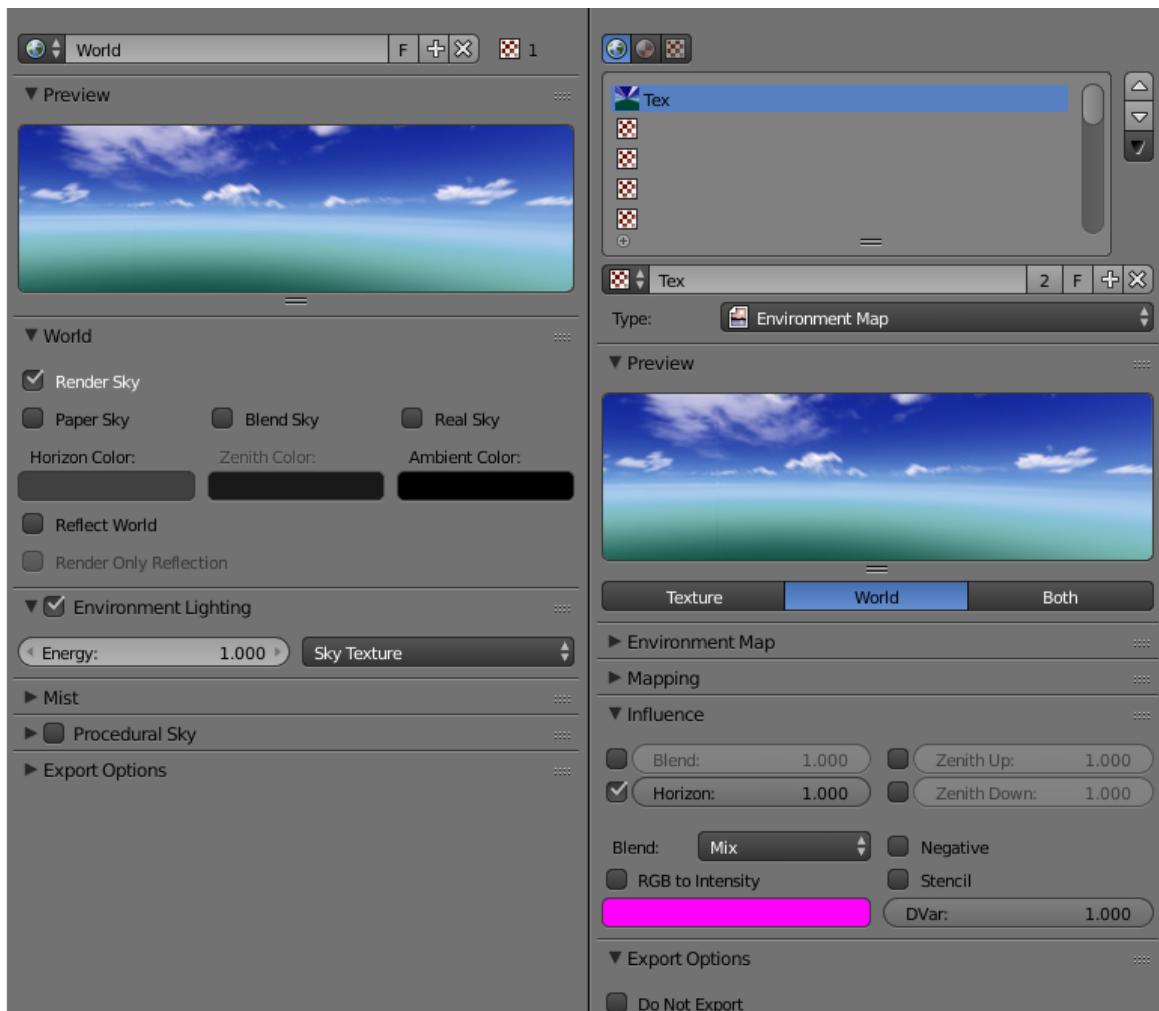
Activation

Create a world texture of Environment Map type. Select the Export Options > Sky Texture Usage > SKYDOME option. Enable World > Render Sky under the World tab.

Note: The behavior of the texture is intentionally made as close as possible to the Blender internal render. That's why the texture may not be displayed upon its default settings. In order to make the texture visible, enable the Influence > Horizon checkbox on its panel and set the Horizon value to 1.0.

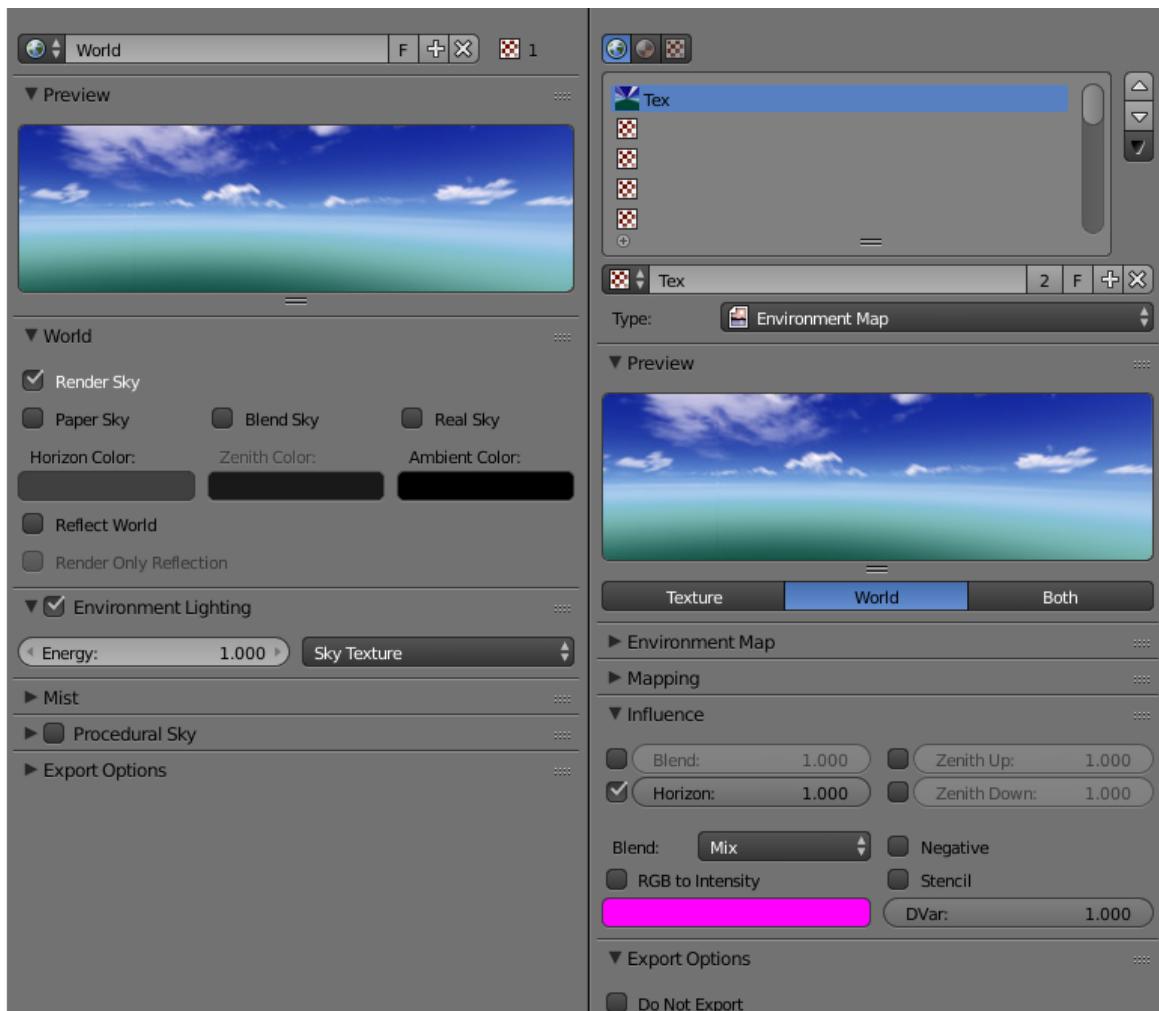
Note: To imitate [environment lighting](#) you can select the Export Options > Sky Texture Usage > ENVIRONMENT_LIGHTING option. Also, you should select the corresponding option in the world settings: Environment Lighting > Sky Texture.

To use the world texture both for skydome and for environment lighting, select Export Options > Sky Texture Usage > BOTH.



Additional Settings

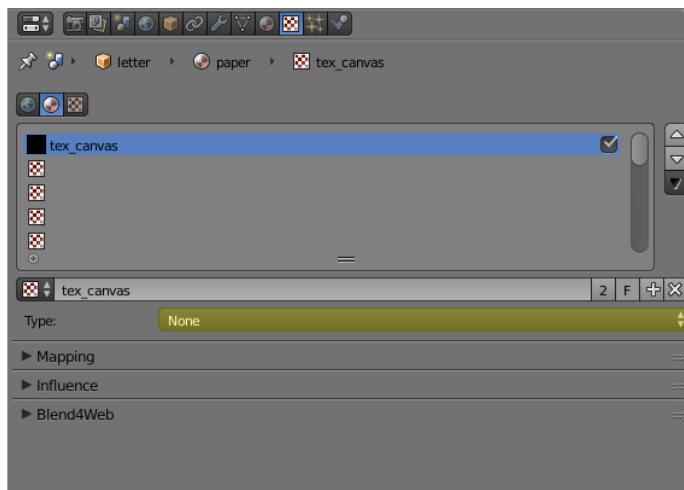
The engine also supports parameters from the world texture's Influence panel which are used for sky rendering. Mixing of the world texture with color depends on the World > Horizon Color and World > Zenith Color parameters, as well as on the Paper Sky, Blend Sky and Real Sky options. All mixing options are supported (Mix, Add, Multiply etc).



Note: The Influence panel parameters only affect the sky rendering. They do not affect [environment lighting](#) by any means.

Special Texture Types

In order to use such textures, select None type under the Textures tab.



On the Textures > Export Options panel, you can set up properties for these textures:

Export Options > Source Type Select texture type: Scene - for rendering a 3D scene into the texture, Canvas - for using <canvas> HTML element and None - for indicating of its absence.

Export Options > Source ID The name of the scene which will be rendered into the texture (for Scene), or ID of the <canvas> HTML element (for Canvas).

Export Options > Source Size Texture resolution.

Export Options > Extension Texture coordinates interpretation mode. Default is Repeat.

Export Options > Enable Mipmapping Enable mipmapping for the Canvas texture. Enabled by default.



3D scene

A 3D scene's real-time rendered image can be used as a texture by an object from another scene ("main" scene).

1. Create an additional source scene.
2. Rename it for convenience.

3. Create a World.
4. Add the objects you need.
5. Setup the camera view.
6. Create a UV map for the target object in the main scene.
7. Set the None type for a texture of the target object.
8. Select the Scene type in the Export Options > Source Type menu.
9. Specify the name of the source scene in the Export Options > Source ID field.
10. Set the texture size in the Export Options > Source Size field (in pixels).



The engine also supports the cyclic rendering of scenes to each other.

Note: A project should contain at least one scene which is not rendered by any other scenes.

Canvas

A <canvas> HTML element can be used as a texture. It can be modified via API.

Set the None type for the texture of the target object on the main scene, and select the Canvas type in the Export Options > Source Type menu. Set the texture size in the Export Options > Source Size field (in pixels).

Use the textures module to handle such textures. See the example below.

```
var m_tex = require("textures");
...
var obj = m_scenes.get_object_by_name("NAME");
var ctx = m_tex.get_canvas_ctx(obj, "TEXTURE_NAME");
...
// operations with canvas context
...
m_tex.update_canvas_ctx(obj, "TEXTURE_NAME");
```

Use `get_canvas_texture_context()` to obtain a context - this method requires the “`canvas_id`” identifier which should be defined in Blender. After `operations with the context`, the `update_canvas_ctx()` function has to be called which will render modifications of the “`canvas_id`” element.

Note: If one Canvas type texture is assigned in Blender to several different objects, then after engine loading it still will be one texture and not several different ones. Any changes applied to it will be applied to all objects using this texture, which can be useful for optimization purposes. In case this effect is not needed, you should assign different textures in Blender or use `deep copy` after engine startup.

Particle System. Fluids

Table of Contents

- Particle System. Fluids
 - Usage
 - * Necessary Steps
 - * Recommended Additional Settings
 - Setup
 - * Basic Settings
 - * Emission Settings
 - * Direction Settings
 - * Rotation Settings
 - * Physics Settings
 - * Rendering Settings
 - * Supported Settings for Force Field Influence
 - * Engine Specific Settings
 - Textures in Particle Systems
 - * Textures of the Particle's Material
 - * Textures of Particle Systems

The particle system is intended to visualize phenomena which are caused by the movement of numerous small objects such as smoke, fire, water splashes and other.



A particle system requires an emitter - an object which defines the location and the direction of the outgoing particles flow.

Usage

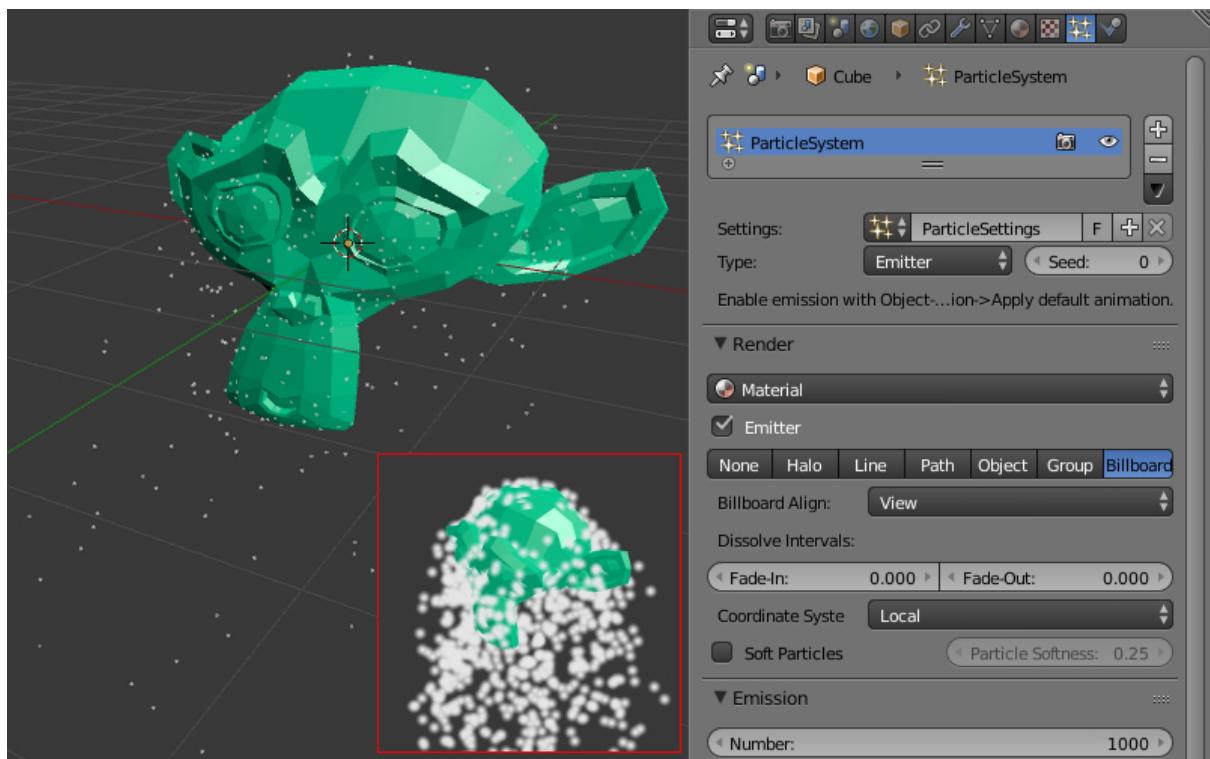
Necessary Steps

1. Add a mesh emitter to the scene.
2. Create a material for particles on the emitter, for example of the Halo type. The Surface material type with a mandatory diffuse texture is also supported.
3. Add a particle system on the emitter.
4. Initiate the engine playback. Two options are available:
 - “cyclic emission” - enable the Emission > Cyclic emission option for the particle system and the Animation > Apply Default Animation for the emitter.

- “non-cyclic animation” - enable the Animation > Apply Default Animation option for the emitter.

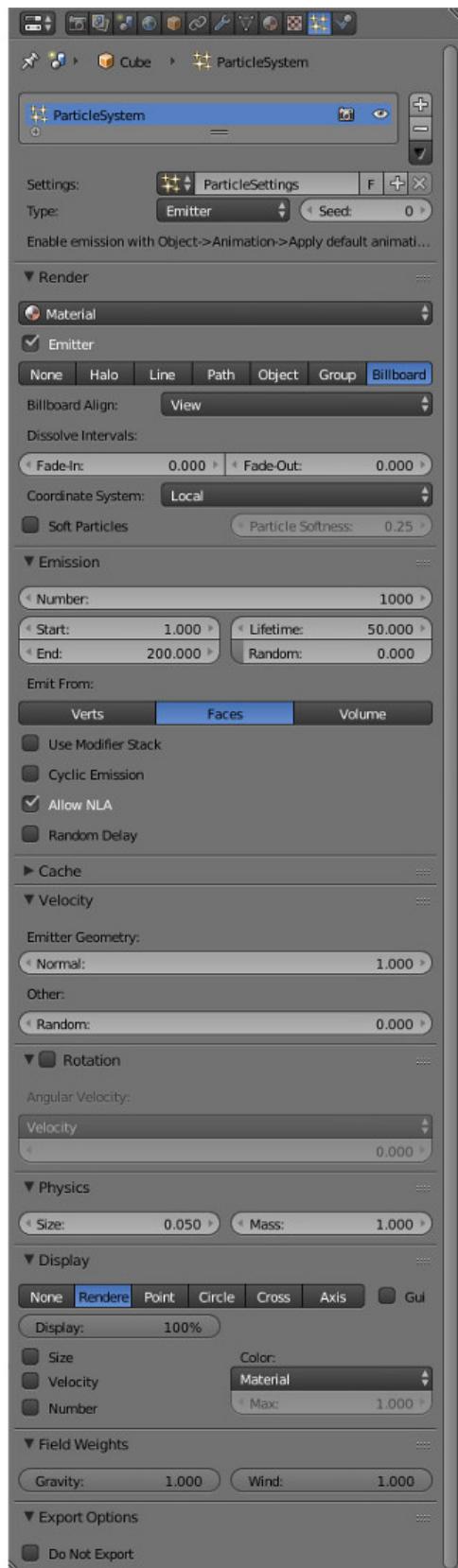
Recommended Additional Settings

1. Set the Add transparency type for the particles’ material.
2. Disable emitter rendering if needed using the Particles > Render > Emitter checkbox.
3. If an emitter is required on a scene use additional materials for it. In this case select the particles’ material in the Particles > Render > Material menu on the particles settings panel.
4. If the Surface material type is used it is required to add a diffuse texture (normally with the alpha channel) to this material. Select UV in the Mapping > Coordinates menu. Make sure that the emitter’s mesh has a UV layer.



Setup

The particle system parameters can be set up under the Particles tab. Multiple particle systems per emitter are supported.



Basic Settings

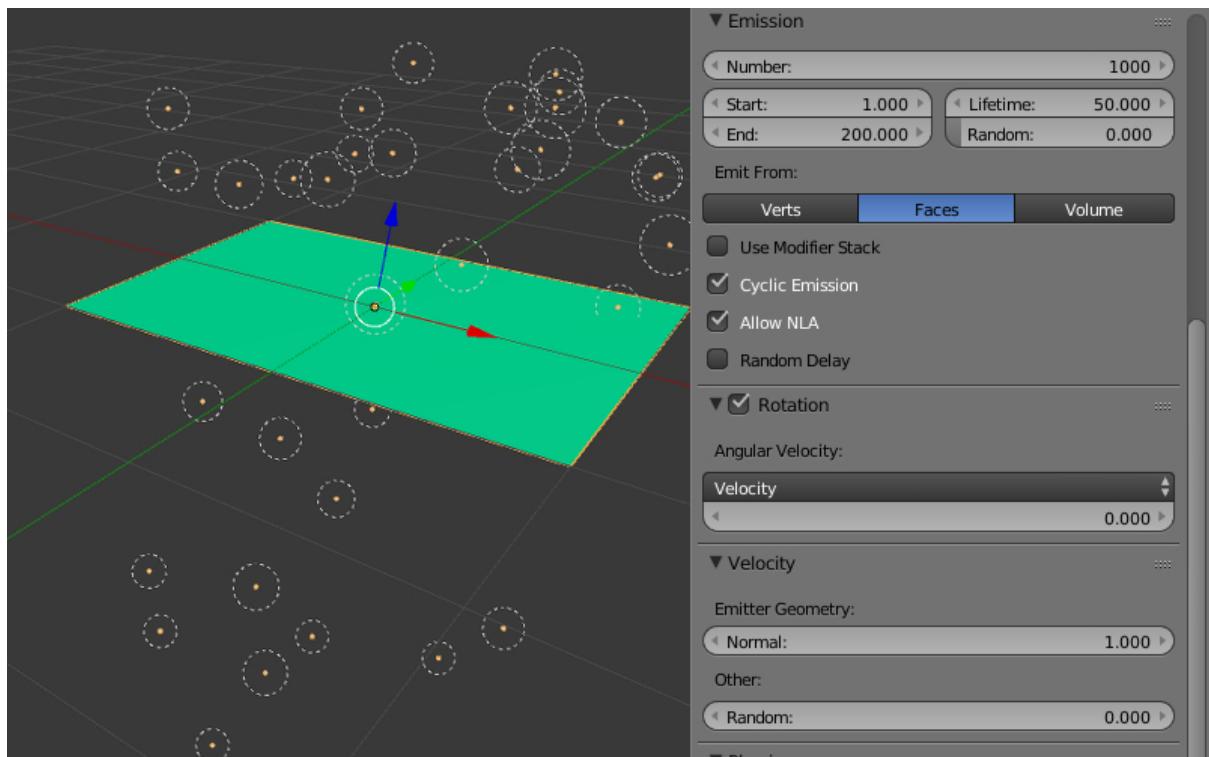
Name Particle system name. The default name is “ParticleSystem”.

Settings Reference to the settings datablock of the particle system. The datablock settings can be shared between different particle systems.

Type Particle system type: Emitter or Hair. Hair particle systems can be used to create numerous copies of an object (so called instancing). The default is Emitter.

Seed Index in the table of random numbers which are used for particle system generation. The default value is 0.

Emission Settings



Emission > Number Number of particles. The default value is 1000.

Emission > Start The first frame after which the emission of particles starts. The default value is 1.0.

Emission > End The last frame after which the emission of particles ends. The default value is 200.0.

Emission > Lifetime The lifetime of particles measured in frames. The default value is 50.0.

Emission > Lifetime > Random The random factor for the lifetime. The default value is 0.0.

Emission > Emit From Emission source type. The following types are supported: Verts (emit from vertices), Faces (emit from polygons). The default is Faces.

Emission > Cyclic emission The option enables the cyclic emission mode. It can be used for permanent effects (such as smoke, burning, water splashes). It is recommended to set the Emission > Start value to zero. Disabled by default.

Emission > Random Delay The option enables a random emission time for particles. Disabled by default.

Emission > Emit From > Distribution Emission distribution settings: Jittered, Random, Grid. Ignored by the engine. Internally the engine always uses Random distribution. The default is Jittered.

Direction Settings

Velocity > Emitter Geometry > Normal Factor influencing the emission along the emitter's mesh normals. The default value is 1.0.

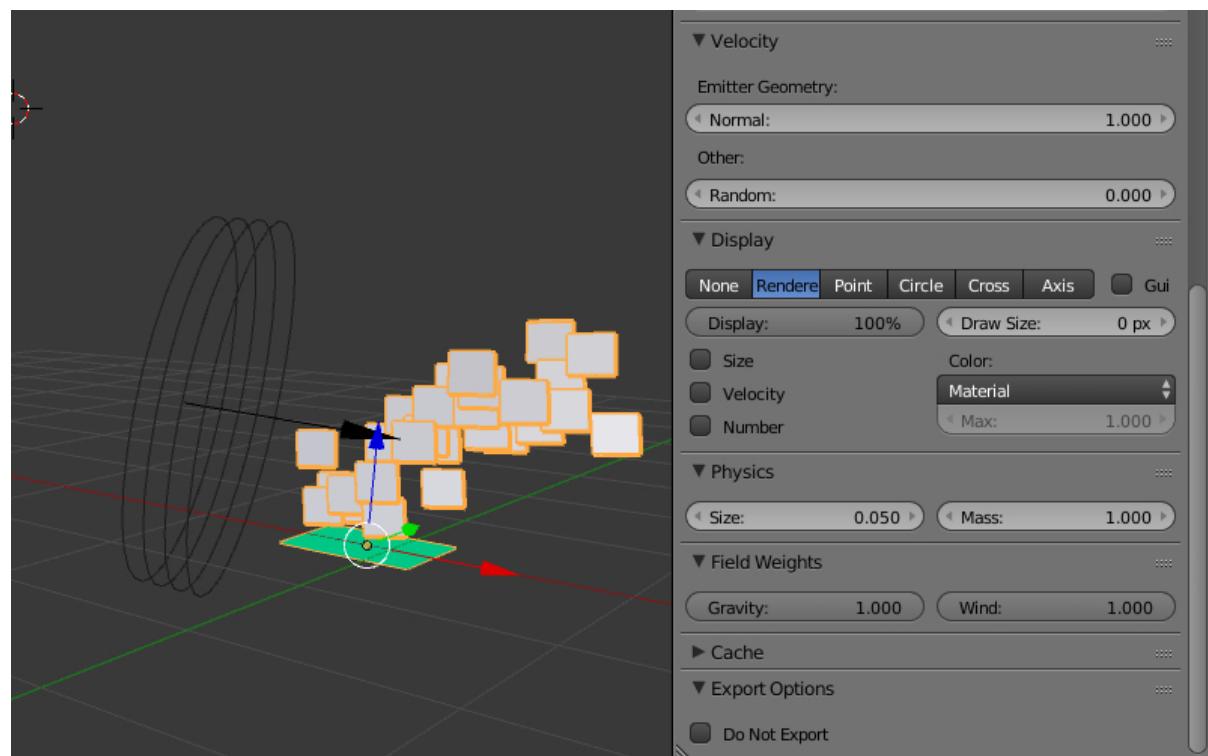
Velocity > Other > Random Factor of randomization for emission direction. The default value is 0.0.

Rotation Settings

Rotation > Angular Velocity > Mode Mode for particle billboards self-rotating. Hair particle system supports all the types of rotation and Emitter supports only Velocity.

Rotation > Angular Velocity > Factor Factor of rotation velocity for particle billboards. The default value is 0.0.

Physics Settings



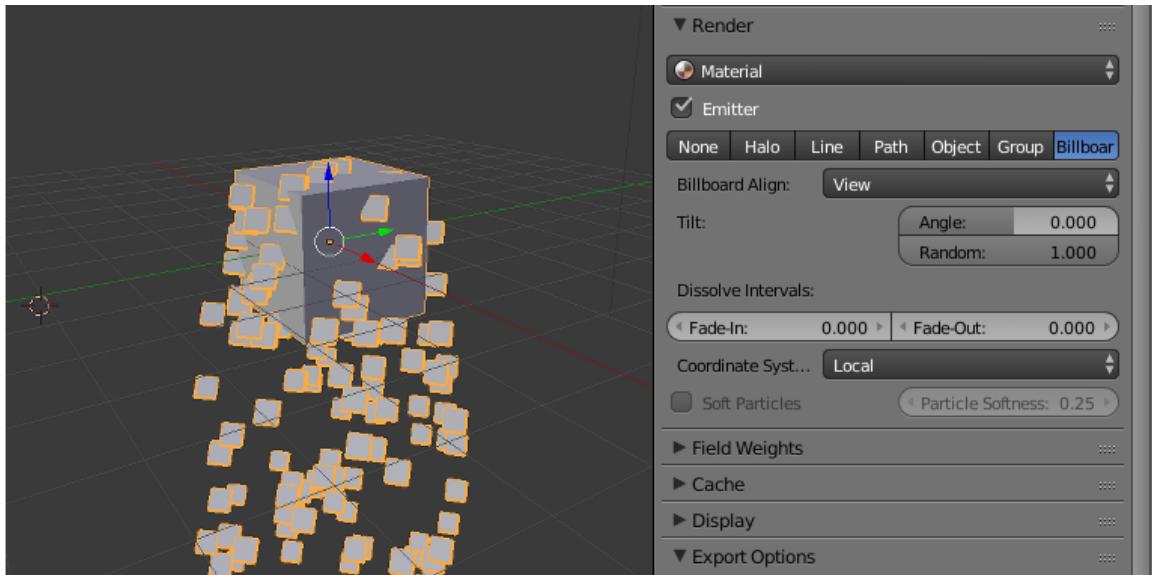
Physics > Type Physics calculation type: No, Newtonian, Keyed, Boids, Fluid. Ignored by the engine. Newtonian physics is always used. The default is Newtonian.

Physics > Size Particle size. The default value is 0.05.

Physics > Mass Particle mass. Affects interaction with force fields (such as wind). The default value is 1.0.

Physics > Forces > Brownian Exported but not used by the engine.

Rendering Settings



Render > Material Menu for selecting the particle's material. Used for referencing to the particle' material in case multiple materials are used by the emitter. The default value is Default Material.

Render > Emitter Enables emitter rendering on the scene. Enabled by default.

Render > Type Particle rendering mode: None, Halo, Line, Path, Object, Group, Billboard. The engine supports the Object and the Group modes which are used for objects and groups instancing respectively. Other modes are ignored. It is recommended to use the Billboard mode for convenient display of billboards. The default is Halo.

Render > Billboard Align The way billboards are oriented: View - follow the camera, XY plane, YZ plane, ZX plane - align to the corresponding plane (in the world coordinate system of Blender). The default is View.

Render > Tilt This group of parameters controls the initial tilt of the billboard planes. It consists of two parameters:

Angle Rotation angle of the billboard planes. Its value can vary from -1 to 1. The value of 1 rotates by 180 degrees (i.e. turns the billboard upside down). Set to zero by default.

Random Random variation of tilt. Its value can change from zero to 1 and is set to zero by default.

Render > Dissolve intervals > Fade-in* and Fade-out Starting and ending intervals (measured in frames) for gradually increasing and decreasing the particles' transparency.

Render > Coordinate System Coordinate system of emitting particles: Local - use local coordinate system of the emitter object, World - use world coordinate system.

Render > Soft Particles Support for soft particles, billboards of which smoothly dissolve when contacting with surfaces. Use the Particle Softness slider to tweak this effect.

Supported Settings for Force Field Influence

Field Weights > Gravity Gravity influence factor (Earth's attraction). The default value is 1.0.

Field Weights > Wind Wind influence factor. A Wind force field source should be present (can be added using Add > Force Field). A particle system is also influenced by the wind direction and strength. The default value is 1.0.

Engine Specific Settings

Export Options > Do not export Don't export.

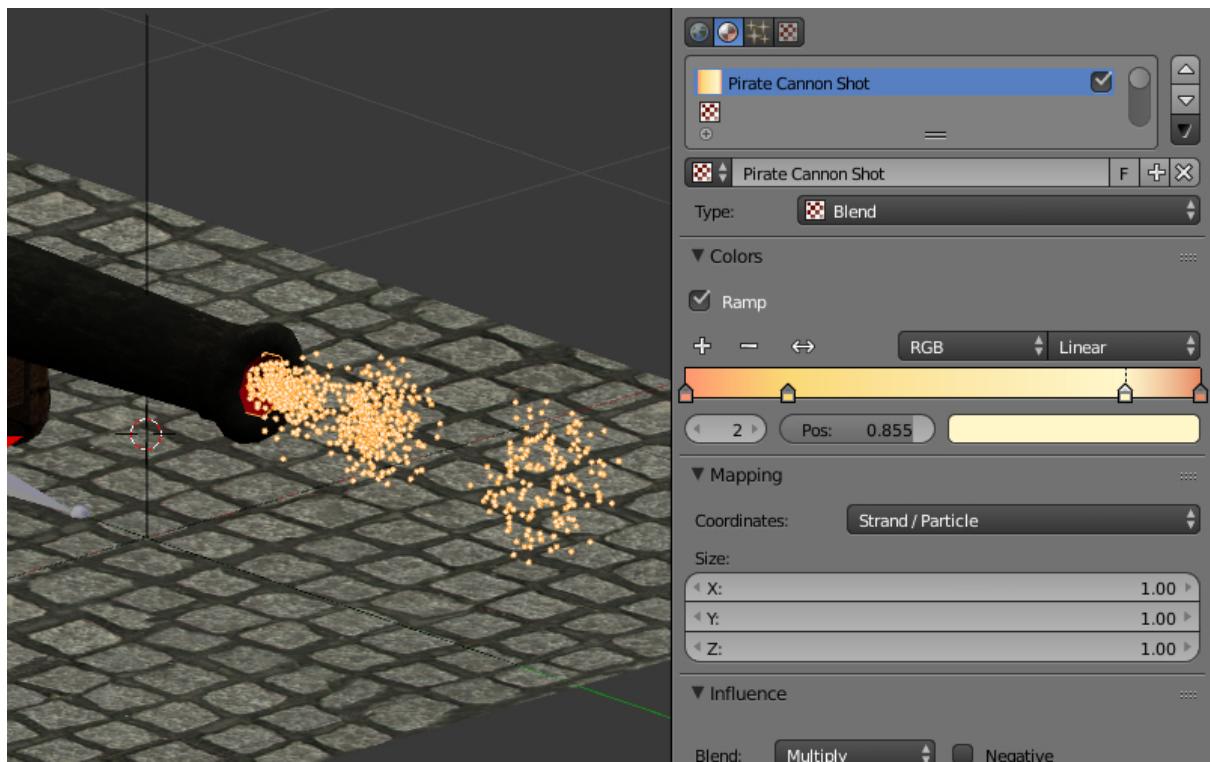
Textures in Particle Systems

Particle systems support both standard and node materials. Supported material types are Surface and Halo. All additional settings of the Halo type materials, such as Rings, Lines and Star Tips, are also supported.

Textures of the Particle's Material

For the Surface particle's materials it is required to have a diffuse texture (normally with an alpha-channel). In the Mapping > Coordinates menu choose the UV option. Make sure that the emitter's mesh has a UV layer.

For the Halo particle's materials it is possible to use a Blend texture with a Linear gradient. In the Mapping > Coordinates menu choose the Strand / Particle option. It is required to enable Ramp on a texture.

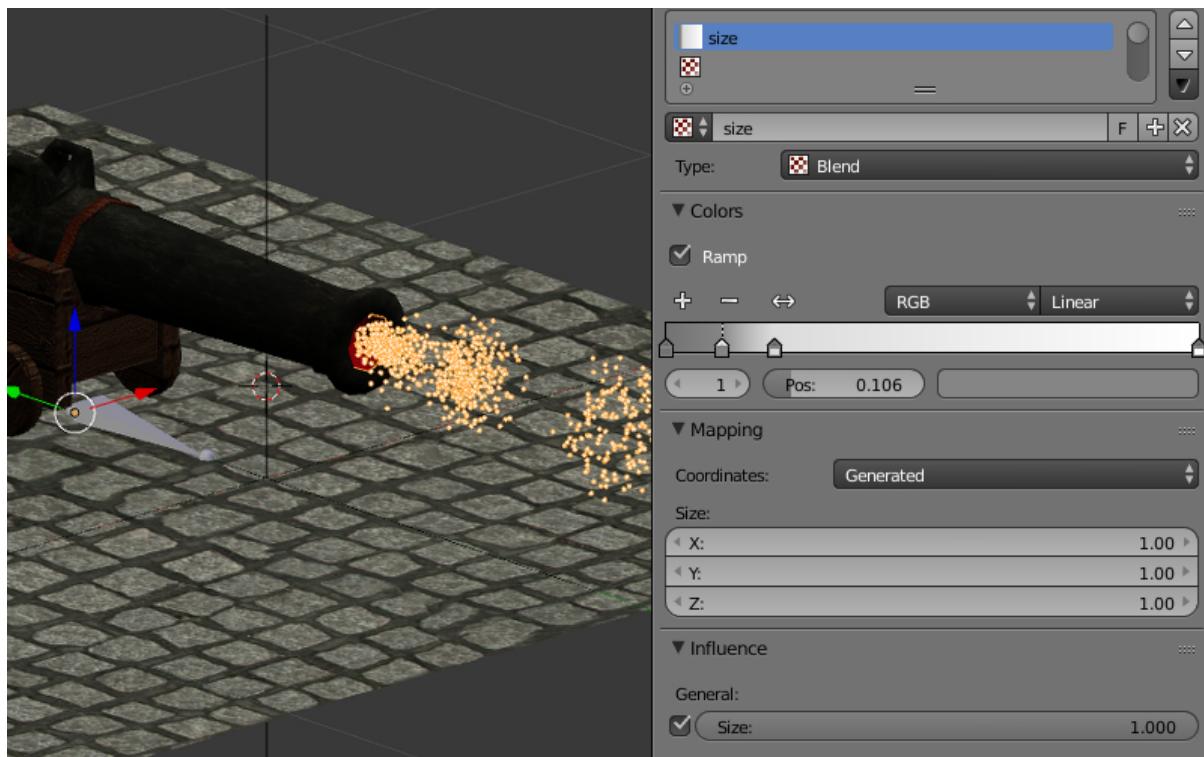


Textures of Particle Systems

Textures can also be used for setting up the behavior of particle systems. Unlike textures for particle materials such textures belong to the particle system datablock, not to the material datablock. To create a texture for the particle system it is required to go from the Particles tab to the Textures tab and then to click the New button.

The only supported type of textures is Blend with a Linear gradient. Ramp should be enabled on the texture. Up to 4 gradient control points are supported.

On the Influence panel choose the parameter which is influenced by the texture. At the moment the only supported parameter is Size.



The result of using gradient textures on the particle material and the particle system:



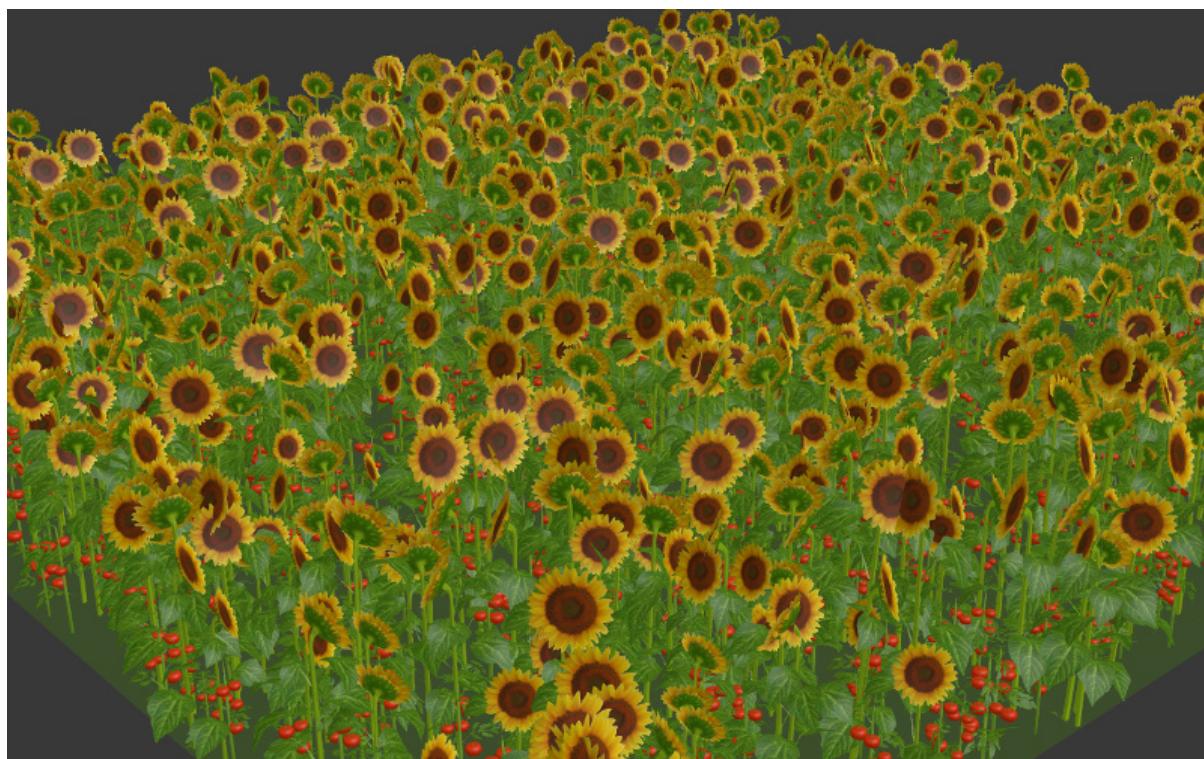
The original model was taken from here

Particle System. Instancing

Table of Contents

- Particle System. Instancing
 - Particle System Setup
 - * Activation
 - * Display settings
 - * Dynamic grass setup
 - * Inheritance settings
 - Grass
 - Tree Leaves

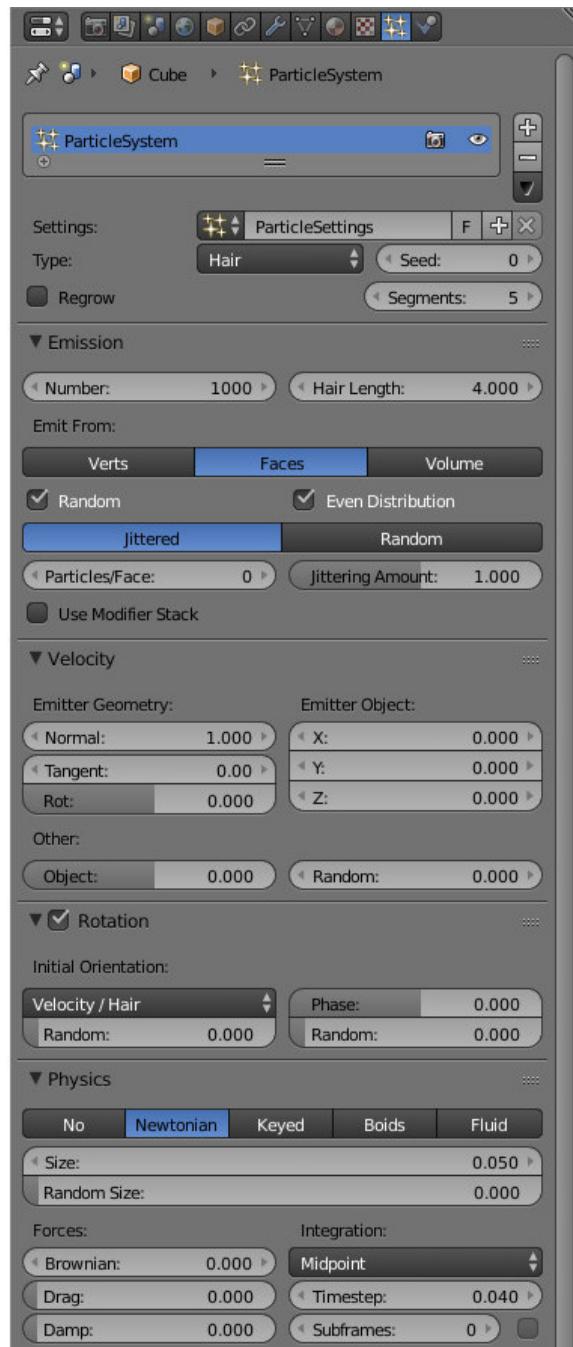
A particle system can be used to create multiple object copies (so called instancing). This technique simplifies scene authoring and also reduces loading time and memory consumption as compared to the using of single objects.

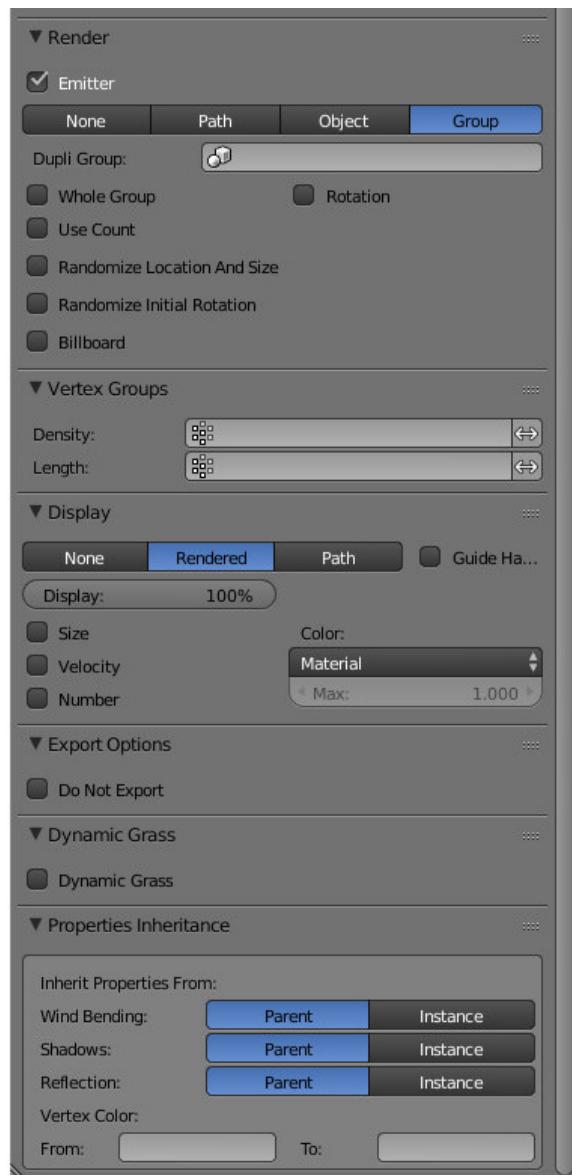


Using particle systems for instancing has some limitations though:

- Movement and animation of objects inside a particle system is not allowed.
- Parenting is not possible among the objects inside a particle system, except for dupli-groups.
- Instancing of non-mesh objects is not possible.

Particle System Setup



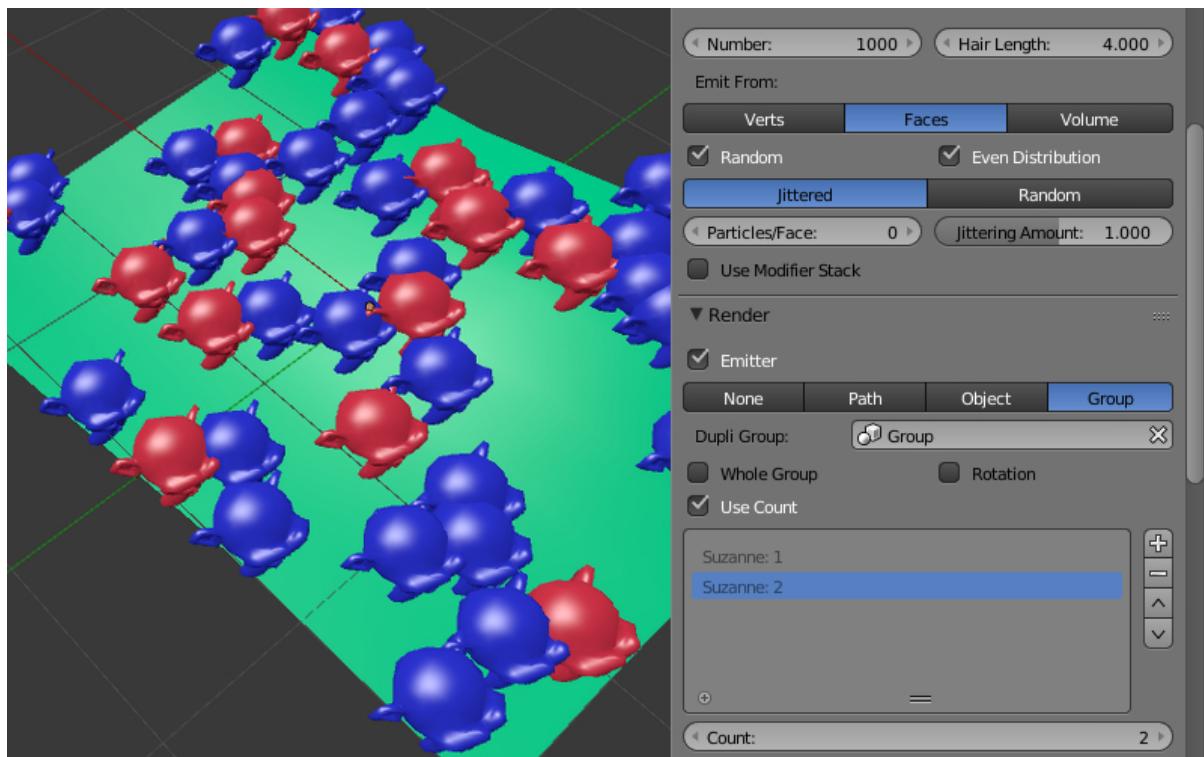


Activation

1. Create a particle system of the Hair type on the emitter.
2. On the Render panel select the Object (or the Group) rendering type.
3. In the Dupli Object field (or in the Dupli Group field) select the object (or the object group) for instancing. Both local and linked objects (or groups) are supported.

Recommended Additional Settings

1. In order to display correct sizes in the viewport, set the Emission > Hair Length and Render > Size parameters to 1.0.



Display settings

Render > Use Count

The option is available for groups of particle objects. When enabled, the interface for setting the relative number of objects in a group becomes visible. The engine does not reproduce the exact positions of objects of certain types.

Render > Randomize Location and Size

The option enables randomization for the location and the size of the objects. If enabled, the engine generates random coordinates and size (limited to the ±25% range) for the particle objects. If disabled, the exact coordinates and sizes of the particle objects are exported and used.

Render > Randomize Initial Rotation

This option randomizes rotation of the objects relative to the axis defined by Rotation Type. If enabled, the engine generates random rotation angles for the particle objects. If disabled, the rotation is taken from the Rotation panel.

Render > Rotation Type

An Axis of random object rotation (the property is available when Render > Randomize Initial Rotation is enabled). There are two options:

- Z axis - the objects are turned randomly around the vertical Z axis
- Random axis - the objects are turned randomly around a random axis

The default is Z axis

Render > Rotation Strength

Coefficient which defines the range of random rotation angles - counting from the direction towards the camera. Available when the Render > Randomize Initial Rotation checkbox is enabled. Examples:

- Rotation Strength = 1 - the angles will lie within the $[-\pi, \pi]$ range
- Rotation Strength = 0.5 - the angles will lie within the $[-0.5 \cdot \pi, 0.5 \cdot \pi]$ range
- Rotation Strength = 0.1 - the angles will lie within the $[-0.1 \cdot \pi, 0.1 \cdot \pi]$ range

The default value is 1.

Render > Billboard

Enables billboarding for particles. Disabled by default.

Render > Billboard Type

Billboarding type. The option is available when the Render > Billboard option is enabled. Three types are available:

- Basic - simple one-sided billboarding: particles will be turned with their front to the observer
- Random -Random - random two-sided billboarding: particles will be more often turned with their front or rear to the observer and less often with their side; also there will be a small random turn; this model is designed specially for grass instancing
- Jittered - Jittered - one-sided billboarding with particles wavering along the plane which is turned to the observer; this model is designed specially for instancing of tree leaves

The default is Basic.

Render > Jitter Amplitude

Coefficient which defines the particle oscillation amplitude. Available when the Jittered type is selected from the Render > Billboard Type menu. The bigger this parameter is, the bigger is the oscillation amplitude. The default value is 0.

Render > Jitter Frequency

Particle oscillation frequency in hertz. Available when the Jittered type is selected from the Render > Billboard Type menu. The default value is 0.

Render > Billboard Geometry

Billboard rotation type (the option is available when the Render > Billboard checkbox is set). Two types are available:

- Spherical - spherical billboarding i.e. particles are fully oriented to the observer and their rotation is unlimited;
- Cylindrical - cylindrical billboarding i.e. particles are rotating only around the vertical Z axis;

The default is Spherical.

Dynamic grass setup

Dynamic Grass

The option enables the dynamic grass rendering mode. Disabled by default.

Dynamic Grass > Scale Threshold

Minimum size for dynamic grass particles. Smaller particles will not be rendered. The option is available if Dynamic Grass is enabled.

Inheritance settings

Properties Inheritance > Wind Bending

Inheriting the Wind Bending settings by the particles:

- Parent - inherited from the emitter
- Instance - inherited from the particle object itself

The default is Parent.

Properties Inheritance > Shadows

Inheriting the shadow settings by particles:

- Parent - inherited from the emitter
- Instance - inherited from the particle object itself

The default is Parent.

Properties Inheritance > Reflection

Inheriting the reflection settings by particles:

- Parent - inherited from the emitter
- Instance - inherited from the particle object itself

The default is Parent.

Properties Inheritance > Vertex Color

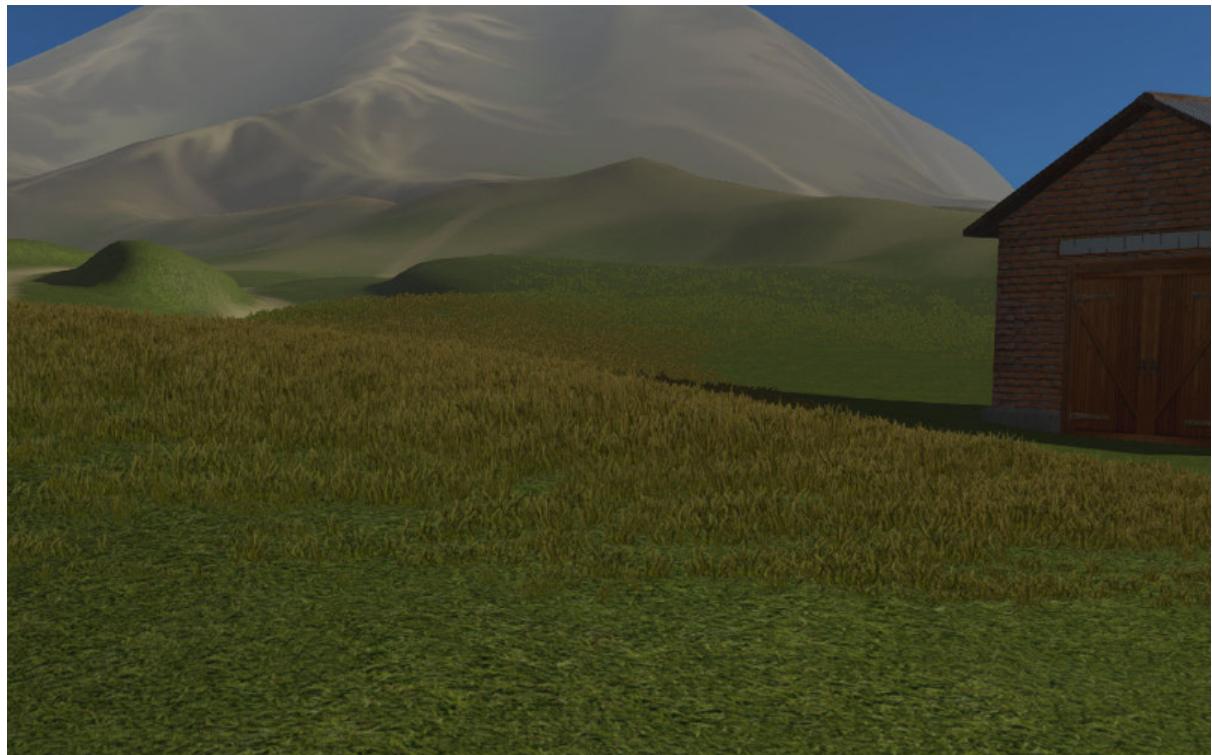
Inheriting the vertex color from the emitter. Contains two fields:

- From - the emitter's existing vertex color name
- To - the particle's existing vertex color name

There is no inheritance by default.

Grass

Instancing of objects can be used for visualizing vast grass. In this case grass is rendered near the camera when it moves through the landscape.



Activation

1. On a separate plane object create a particle system for object instancing. Enable the Dynamic Grass option.
2. Enable the Terrain Dynamic Grass option for the supposed landscape material.

Setup

It is recommended to create a few planes (for example 3) with sizes corresponding to the desired grass cascades (e.g. 100, 150 and 250 meters).

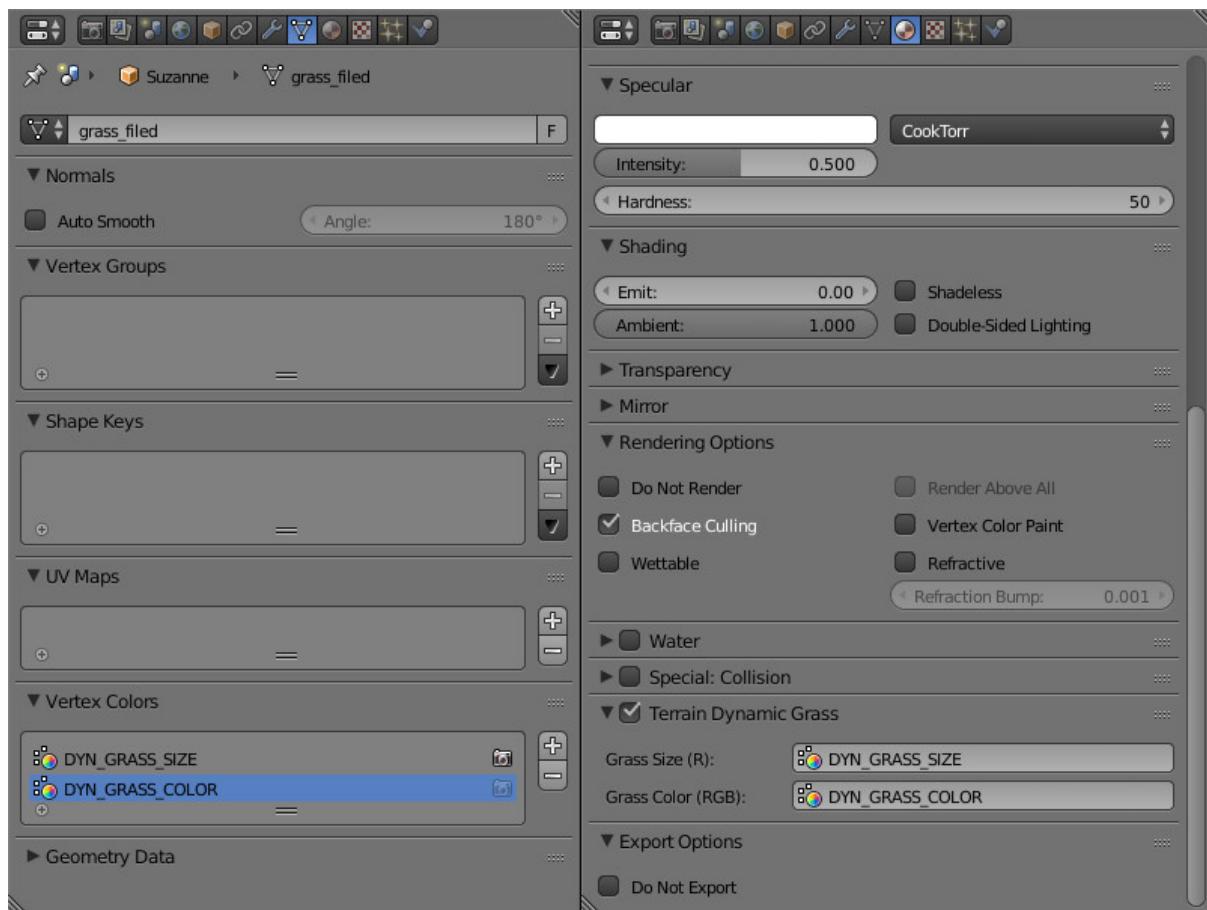
For the landscape's material, the following text fields become active when the Terrain Dynamic Grass option is enabled:

Dynamic Grass Size (R) Vertex color layer name of the landscape mesh which is intended for modifying the grass size. The size (i.e. height) of the grass is defined by gray tints - the brighter color the is the higher is the grass.

Dynamic Grass Color (RGB) Name of the landscape mesh's vertex color layer which is intended for grass tinting. The vertex color is multiplied by the grass material color. The Influence > Blend parameter for the grass material's diffuse texture should have the Multiply value.

Vertex color layers with such names should exist in the landscape mesh.

It is also recommended to disable rendering of the emitter (the Render > Emitter option).



Tree Leaves

Instancing suits the rendering of tree leaves well and allows to get a better level of detail.



Activation

Performed as described in the Particle system setup -> Activation section (see above). In this case the tree is the emitter and the leaves and the small branches are the particles.

Additionally, the following operations can be performed for the emitter:

- create a vertex group which includes vertices on which the particles will be placed
- create a vertex color layer for the wind bending parameters of the tree and the leaves
- create a vertex color layer to be inherited by the particles (for example it can be used for tinting the particles)

Setup

1. Random rotation settings

If the Initial Random Rotation checkbox is enabled, it is recommended to select the vertical axis for random rotation - Z axis (by using the Rotation Type menu). The Rotation Strength value can be set at will.

2. Billboard settings

It is recommended to enable billboard, to set its type as Jittered (by using the Render > Billboard Type menu) and to make it spherical - Spherical (by using the Render

> Billboard Geometry menu). The Render > Jitter Amplitude and Render > Jitter Frequency values can be set at will.

3. Particle position settings

It is recommended to select the Verts value from the Emission > Emit From menu, and to select the emitter's vertex group (in the Vertex Group > Density field) which defines the positions of particles. Note, that the Render > Randomize Location and Size checkbox should be disabled.

4. Wind effect settings

It is recommended to enable inheritance settings from the emitter - select the Parent in the Properties Inheritance > Wind Bending menu. Then for the emitter on the Object panel enable the Wind Bending checkbox and set up the bending parameters. For a tree, it is enough to specify the Wind Bending > Main Bending > Angle and Wind Bending > Main Bending > Frequency parameters and also a vertex color name for bending in the Wind Bending > Main Bending > Main Stiffness field.

5. Vertex color inheritance settings

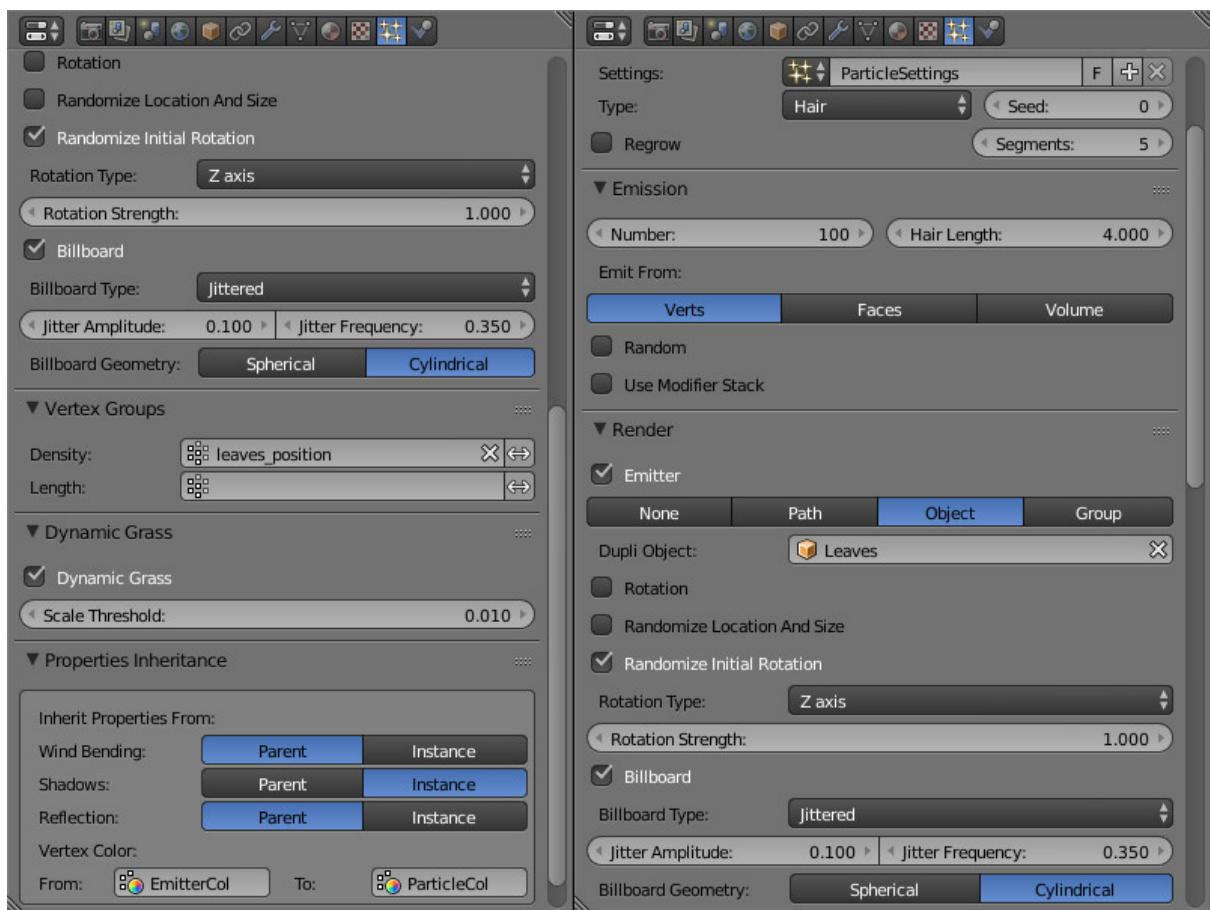
For the emitter's vertex color to be inherited by the particles, it is required to specify both the emitter's vertex color name and the particle's vertex color name in the Properties Inheritance > Vertex Color > From and Properties Inheritance > Vertex Color > To fields respectively. As a result, the color of the emitter's vertex that is closest to the particle (specified in the From field) will be copied and propagated into the particle's To vertex color layer.

The resulting vertex color layer with the name specified in the Properties Inheritance > Vertex Color > To field can be used in the particle's node material for its tinting and for any other effects.

6. Setting up the size of particles via vertex group weights

In order to create dependency between the size of particles and vertex group weights, select the name of the desired vertex group in the Vertex groups > Length field.

The influence can be tweaked by setting weights in the selected vertex group.



Animation

Table of Content

- Animation
 - Animation Control
 - Object Animation
 - Skinning and Skeletal Animation
 - * Baking parameters
 - Vertex Animation
 - * Baking parameters
 - Default Animation
 - Non-Linear Animation
 - * NLA Editor
 - * Controlling via API
 - Audio Source Parametrization
 - Animation of Value and RGB Nodes

In general animation is changing the object's parameters in time. The engine supports the following types of animation:

- Object animation means the transformation of an object as a whole.
- Skeletal Animation, i.e. object deformation using bones. Animation of a standalone armature object is also supported (for parenting to bones).
- Vertex animation. An object's deformations can be recorded as frames and then reproduced in the engine.
- Audio sources parametrization. Speaker's Volume and Pitch can be animated.
- Animation of the Value node output in node materials.
- Wind bending - a procedural animation. Described [separately](#).
- Particle emission. Described in the [corresponding section](#).

Animation Control

There are two ways to control animation in the engine:

1. Automatically, activating the Animation panel and choosing the Behavior parameter in the object's properties. In this case an appropriate animation method will be chosen by the engine and the object's animation playback will start just after a scene is loaded. In case of skeletal animation the action which is assigned to the object in the Action Editor window is played by default.
2. In an application via API using the animation module methods.

It's useful to use the Animation interface for tweaking animation. This is covered in the corresponding section.

Object Animation

The parameters that can be animated are the center coordinates (Location), Rotation and Scale.



Animation keyframes can be added for an object motion in Blender and then reproduced in the engine.

The following keyframe types are supported:

- Location
- Rotation – the Quaternion(WXYZ) or XYZ Euler mode is required.
- Scale – for correct results the scale factor should be the same along all 3 axes.
- LocRot – a combination of Location and Rotation.
- LocScale – a combination of Location and Scale.
- LocRotScale – a combination of Location, Rotation and Scale.
- RotScale – a combination of Rotation and Scale.

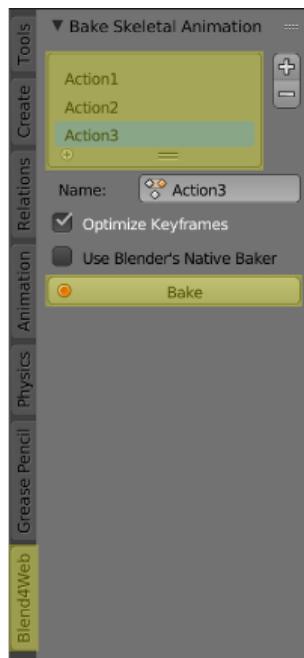
If a mesh object is animated it is required to activate the Force Dynamic Object option on the Rendering Properties panel under the object properties tab.

Skinning and Skeletal Animation



For skeletal animation both a mesh object and an armature object are needed. The four steps should be carried out:

1. Create the object's "skeleton" in the armature object.
2. Assign vertex groups in the mesh object and link them to the bones. This can be performed by weight painting for example.
3. Animate the bones in the pose mode of the armature object. The same keyframe types can be used as for the object animation.
4. When inverse kinematics (IK) or other nontrivial structures are used, an additional step is required to bake the animations (Action datablocks in Blender). Baking can be performed using the Bake Skeletal Animation interface located on the Blend4Web:



Baking parameters

Baking is performed with the armature object selected.

The actors that will be baked are listed in the window with the list of actors. If the list is empty, all available actors will be baked.

Name The current action name from the list of actions being baked.

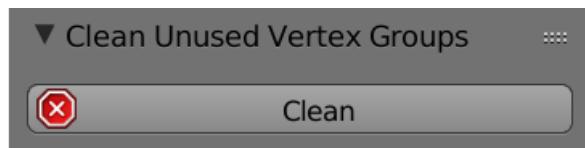
Optimize Keyframes Optimize the animation keyframes after baking. In case of incorrect results it's recommended to turn this option off.

Use Blender's Native Baker Use Blender's animation baker (the Bake Action option) instead of Blend4Web's. Blender animation baker's settings differ from Blend4Web's.

Bake Perform baking. If the process is completed successfully, actions with names of B4W_BAKED_ACTOR_NAME type appear in the scene. These actions can be assigned to the armature object and played back in the engine. It's worth noting that appropriate functioning of such actions in Blender is not guaranteed.

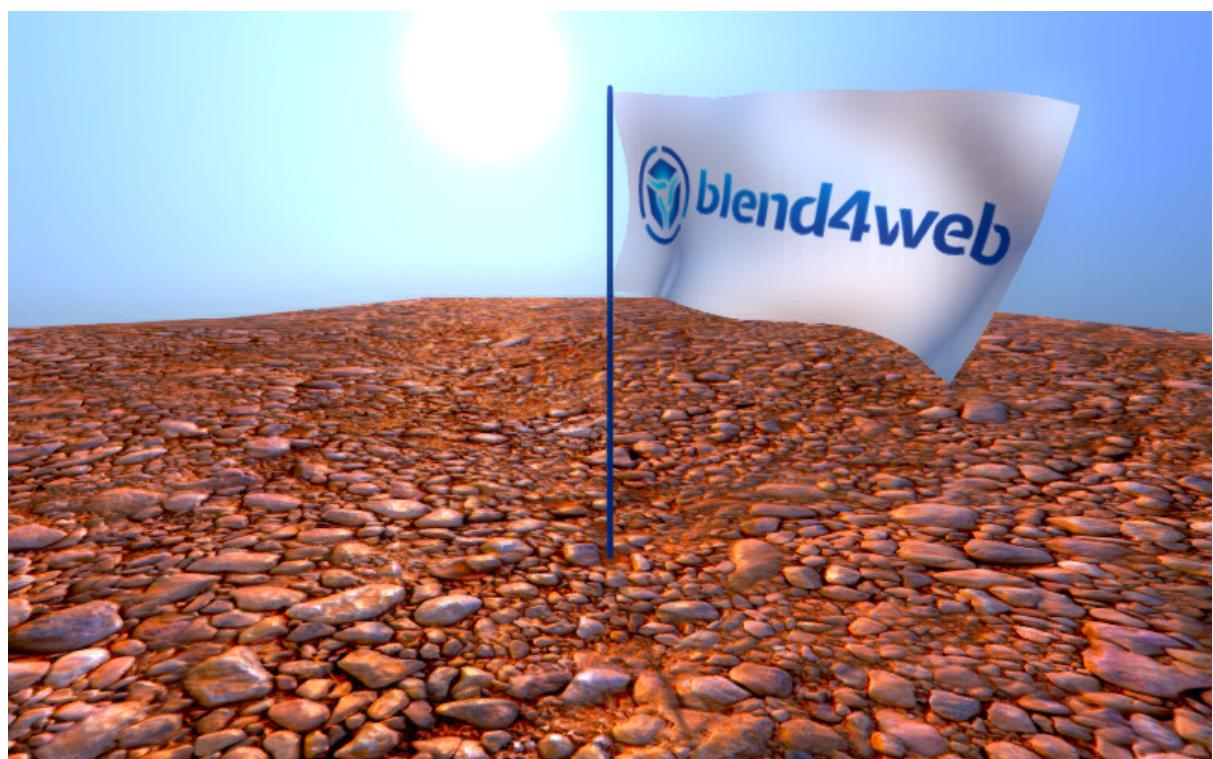
Note: The engine supports up to 4 vertex groups per vertex. If the number of vertex groups exceeds 4, the vertex groups with the most influence are selected. When the scene is loaded, the vertex weights are normalized i.e. their sum is reduced to 1.

To remove vertex groups which are not used by armature, use button Remove Clean Unused Vertex Groups.

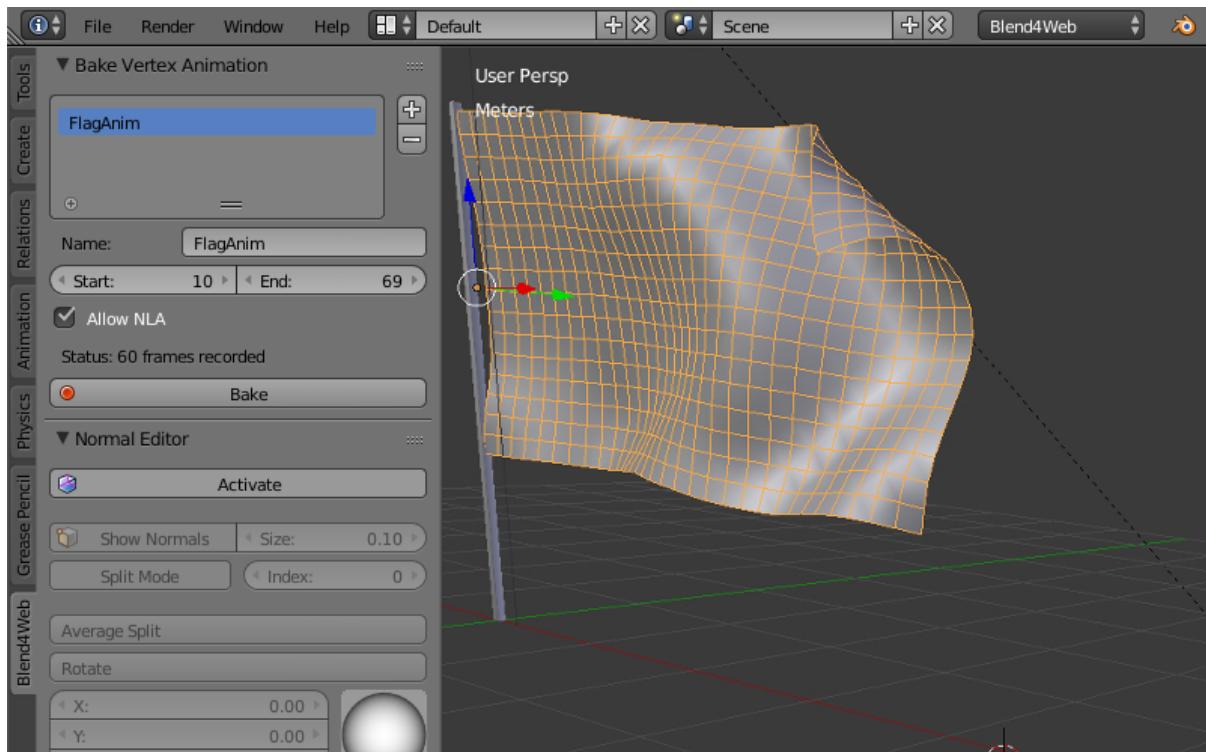


Blend4Web also has initial support of the armature constraints. For now, the only supported constraint type is Copy Transform. It can be used to attach an armature to an object, producing effects such as ragdoll. Support of the other types of constraints will be added in further releases.

Vertex Animation



Allows to record any geometry changes of a mesh object. Note that every vertex animation frame counts as a mesh. It's not recommended to make a long animation for a high-poly mesh, as it can increase the size of the source and exported files significantly and can also slow down the work of the engine. A special tool is used for baking vertex animation - Bake Vertex Animation - located on the Blend4Web tools panel.



Baking parameters

Name The name that will be assigned to the baked animation.

Start The baking will start from this frame.

End The baking will end at this frame.

Allow NLA Allows using [NLA](#) to control the baked animation.

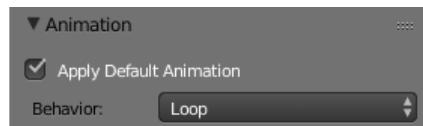
Status In this string, the number of frames of the baked animation is shown. If the baking hasn't been performed yet, the word **Empty** is shown instead.

Bake The animation baking is performed by pressing this button.

Note: For vertex animation to work correctly, the Export Vertex Animation parameter (Export Options section of the Object panel) of the selected object should be enabled.

Default Animation

Blend4Web also has an option to automatically play object's animation. To do it, you need to enable the **Apply Default Animation** parameter in the Animation section of the Object panel of an object you want to animate. Skeletal and object animation types are supported.

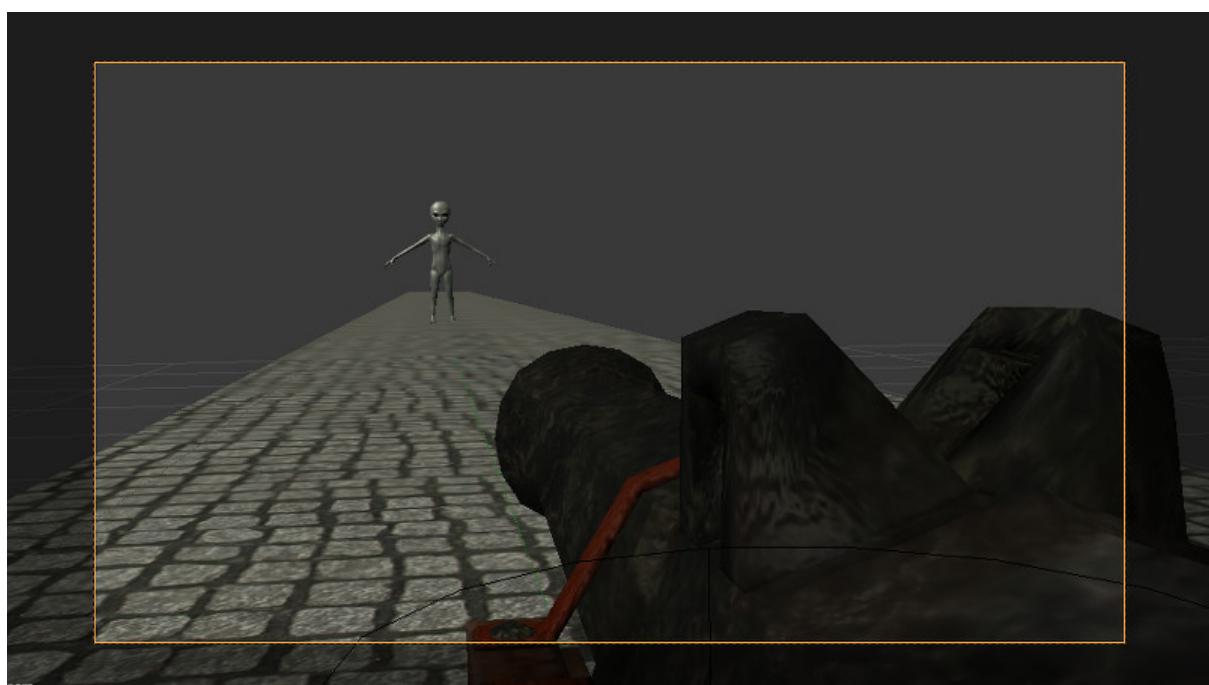


Animation starts to play right after the application startup and plays very similar to the [Play Timeline](#) node, only without an option to set up the start and end markers (instead, it always starts from the first frame of the timeline and ends with the last). You can also set up animation behavior, like in the [Play Animation](#) node.

Non-Linear Animation

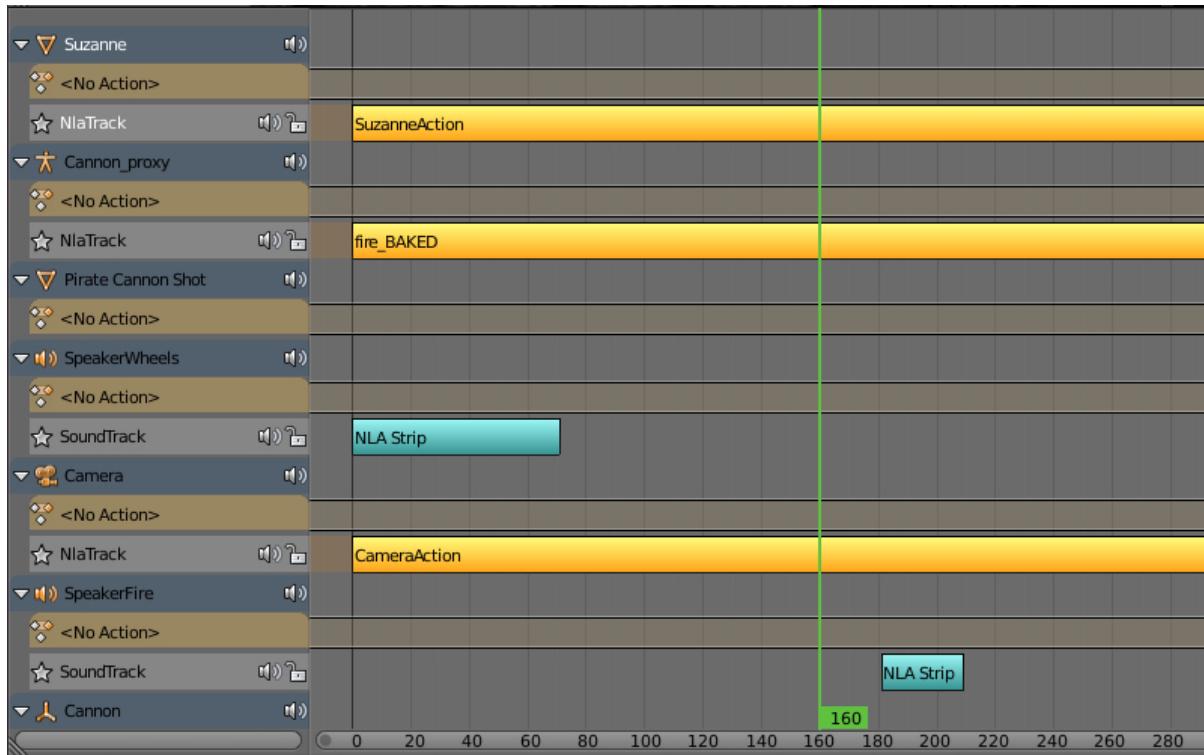
NLA Editor

The Blender's non-linear editor lets us set the scene's behavior in a comfortable way. With its help we can implement simple scenarios. This way coding is not needed for simple scenes and applications.



The engine supports controlling the following entities:

- Any animation the parameters of which can be presented with Actions
- Audio playback
- Particles emission (in the form of a connection with the global timeline)



Usage

1. Activate the NLA panel under the Scene tab.
2. In the NLA Editor set up the required behavior for the scene.
3. Choose the animation time interval on the Timeline panel.

Additional settings

The NLA > Cyclic NLA scene setting activates the cyclic NLA animation mode.

Note: In order to use vertex animation, enable “Allow NLA” option on the [vertex animation panel](#).

Limitations

- A simultaneous playback of different types of animation for the same object is not supported.

Controlling via API

Non-linear animation playback can be controlled via API methods of the nla.js module.

```
// ...
var m_nla = require("nla");
// ...
m_nla.set_frame(150);
// ...
var frame = m_nla.get_frame();
// ...
m_nla.play();
// ...
m_nla.stop();
// ...
```

Please note, that if the [Logic Editor](#) is used, the set_frame, play, stop methods are not available.

Audio Source Parametrization

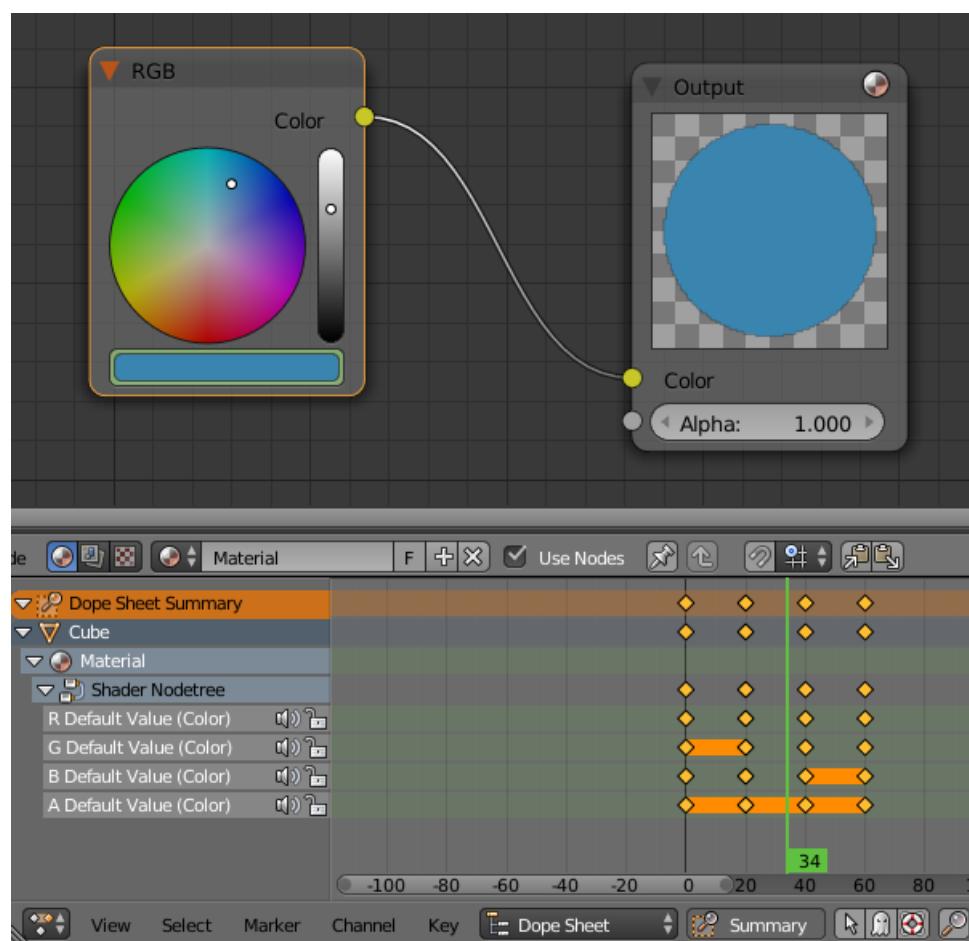
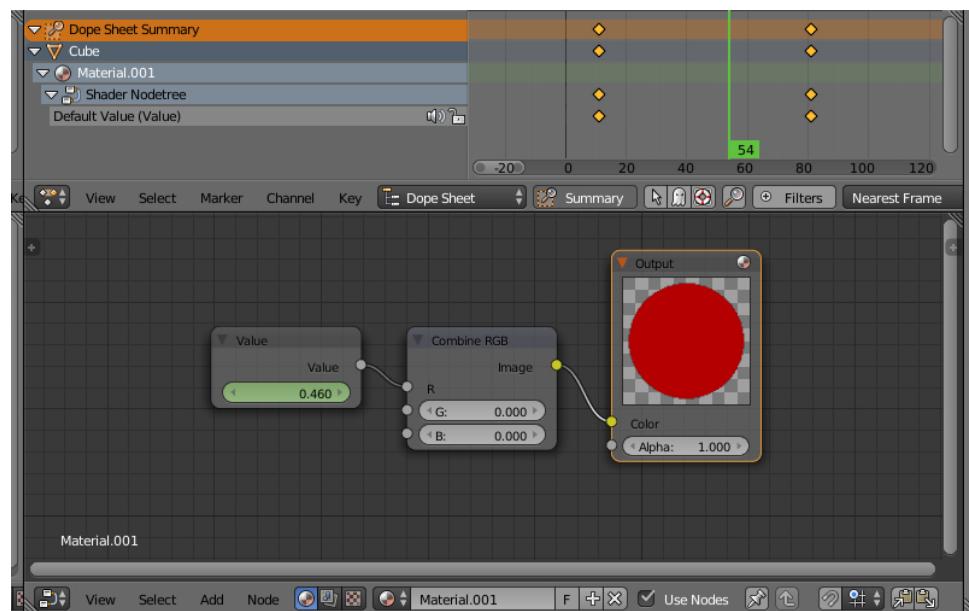
In addition the following animation key types are supported for the speaker objects:

- Volume
- Pitch

Audio sources parametrization in essence follows object animation.

Animation of Value and RGB Nodes

Playback of keyframes inserted on Value and RGB nodes is supported in node materials.



Note: Animation of numerical and color values on other nodes is not supported.

Can be also used to create tracks in the [non-linear animation editor](#). Multiple animated Value or RGB nodes are supported per single material. Values of these nodes can be also modified via API, by using the `set_nodemat_value` and `set_nodemat_rgb` methods of the objects module.

See also:

[Time \(B4W_TIME\)](#)

Outdoor Effects

Table of Contents

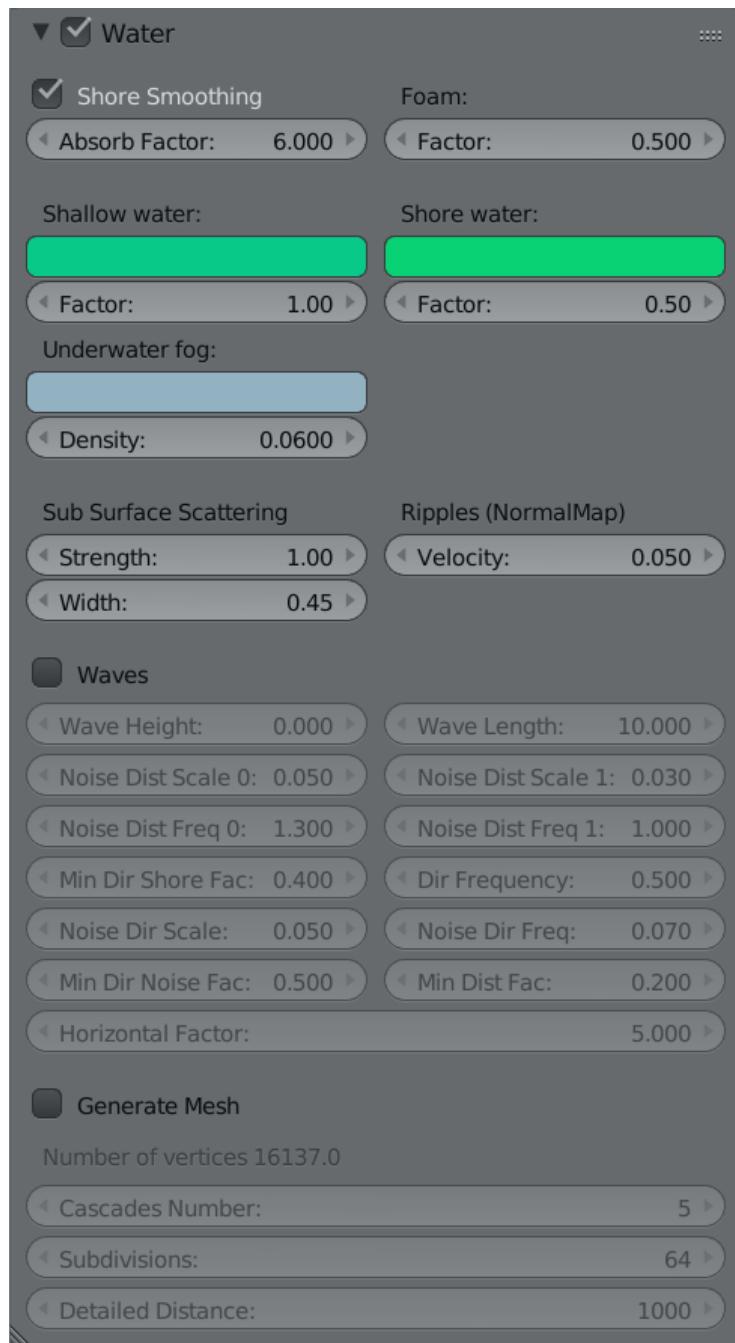
- Outdoor Effects
 - Water
 - * Activation
 - * Basic Settings
 - * Waves Dynamics
 - * Surface Wetting
 - * Reflection and Fresnel Effect
 - * Shoreline Smoothing
 - * Color Gradient
 - * Refraction
 - * Foam
 - * Caustics and Chromatic Aberration
 - * Underwater Environment
 - * Volumetric Waves
 - * Settings for Surface Generation
 - Atmosphere
 - * Scattering
 - * Fog
 - * Time of Day
 - * Stars
 - Wind
 - * Activation
 - * Setting up
 - * Animation of Grass and Tree Leaves

Blend4Web has several environmental effects that can be useful for creating outdoor scenes.

Water

Activation

For the supposed water material activate the Water panel option under the Material tab.



Basic Settings

Transparency It is recommended to enable the gradient transparency (set the Transparency > Type parameter to the Alpha Blend value) and to tweak the Alpha value.

Lighting parameters Lighting parameters for the water material can be set up as described in the [Lighting Parameters](#) section.

Waves Dynamics

Ripples on the water are simulated by normal maps with animated UVs (from 0 up to 4 pieces). For normal map the only shared image is used - the textures differs only by the [Mapping > Size](#) parameters. The water mesh must have a UV layer.

Surface Wetting

Is carried out automatically. To turn the effect on activate the [Wettable](#) option on the needed materials [Rendering Options](#) panel.

Reflection and Fresnel Effect

For the water material both static and dynamic reflection is supported as well as the Fresnel effect. See the [Reflection](#) section.



Shoreline Smoothing

The effect affects the water near the shoreline - it becomes more transparent.

[Water > Shore Smoothing](#) Enable smoothing.

Water > Absorb Factor Light absorption coefficient for the water. The higher it is the more transparent is the water.

Color Gradient

For color gradient the water material must have a texture with the Export Options > Shore Distance Map option enabled. This texture can be generated using the script for baking shoreline parameters.

Shallow Water > Color Shallow water color.

Shallow Water > Factor Shallow water color mixing factor.

Shore Water > Color Water color just at the shore line.

Shore Water > Factor Factor for mixing water color just near the shoreline.

Refraction

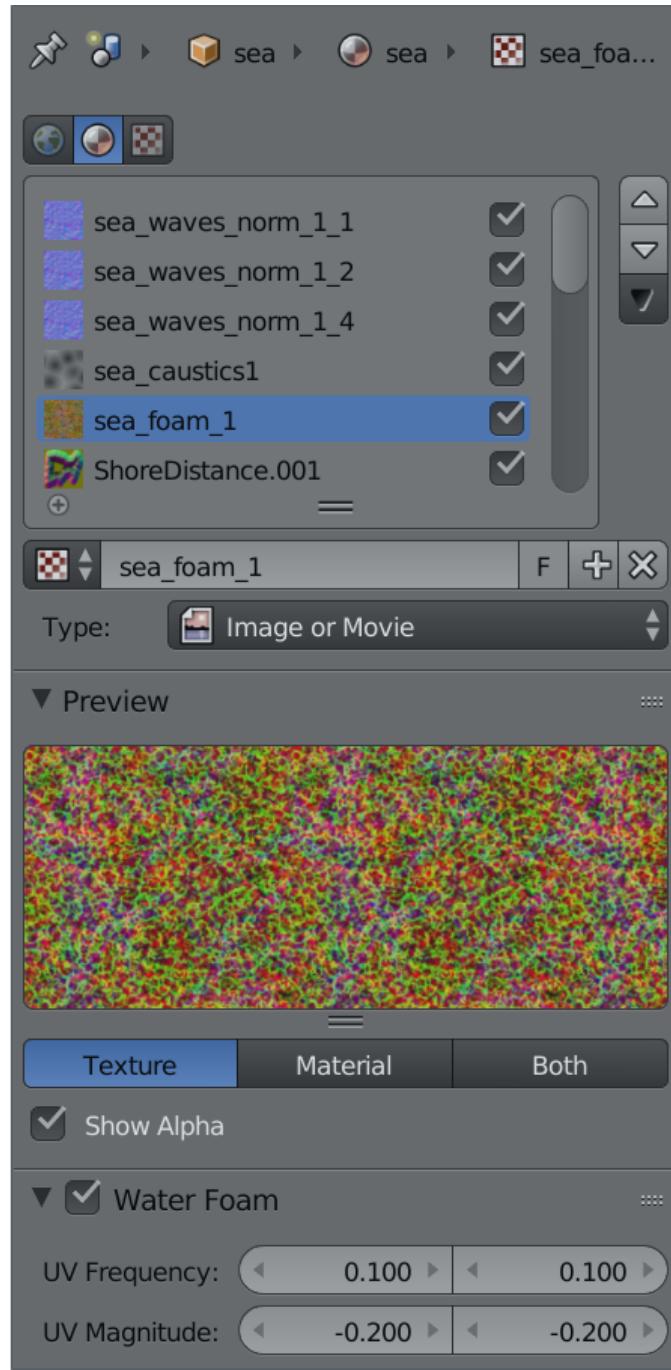
Under the Render tab in the Reflections and Refractions panel set the Refractions option to ON or AUTO.



Foam

Activation

For creating foam add a diffuse texture into the water material slot. Every RGB channel of this image must contain a BW foam texture in it. Then enable the Water Foam panel.



Setting up the Textures

Influence > Color Texture color influence factor. The default value is 1.0.

Water Foam > UV Frequency Oscillation frequency of the animated UV coordinates.
The default value is (1.0, 1.0).

Water Foam > UV Magnitude Oscillation amplitude of the animated UV coordinates.
The default value is (1.0, 1.0).

Setting up the Material

Foam > Factor General influence factor for the foam. The default value is 0.5.

Caustics and Chromatic Aberration

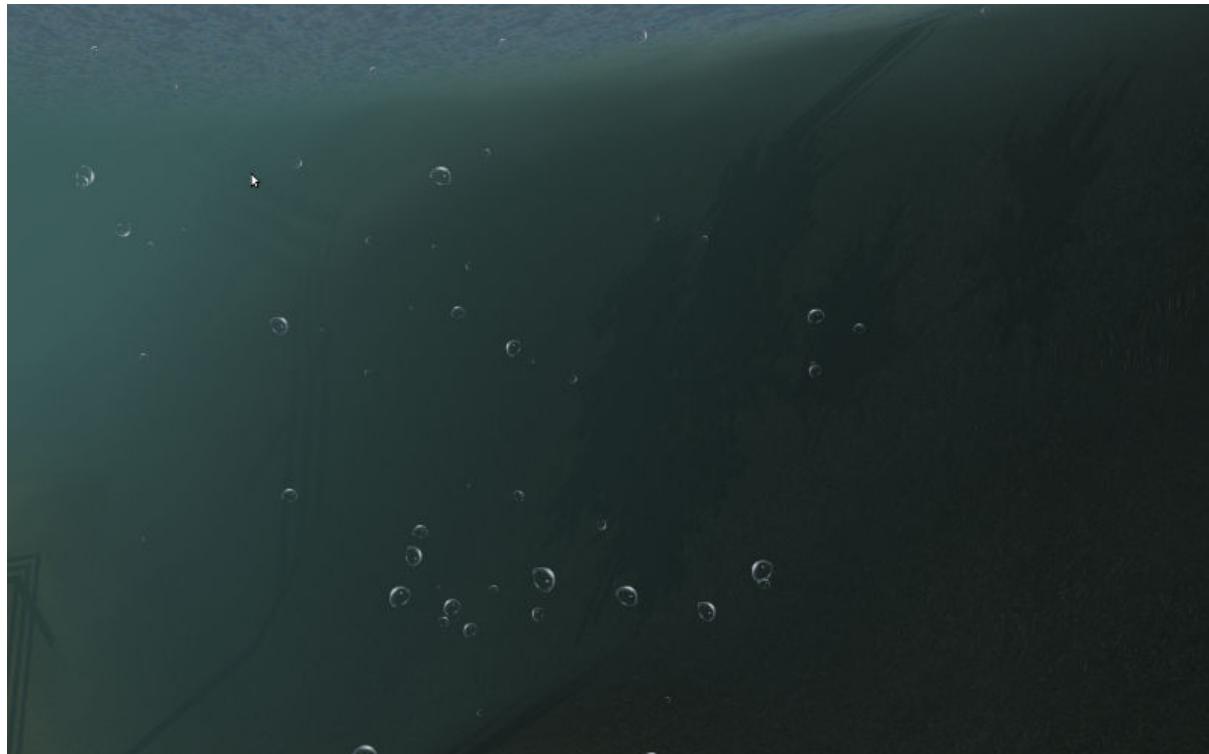
To create the caustics effect turn on the Caustics option on the water material. Also, there has to be at least one Sun on the scene.



Scale Cell size for the procedural texture. The default value is 0.25.

Brightness Caustics influence factor. The default value is 0.5.

Underwater Environment



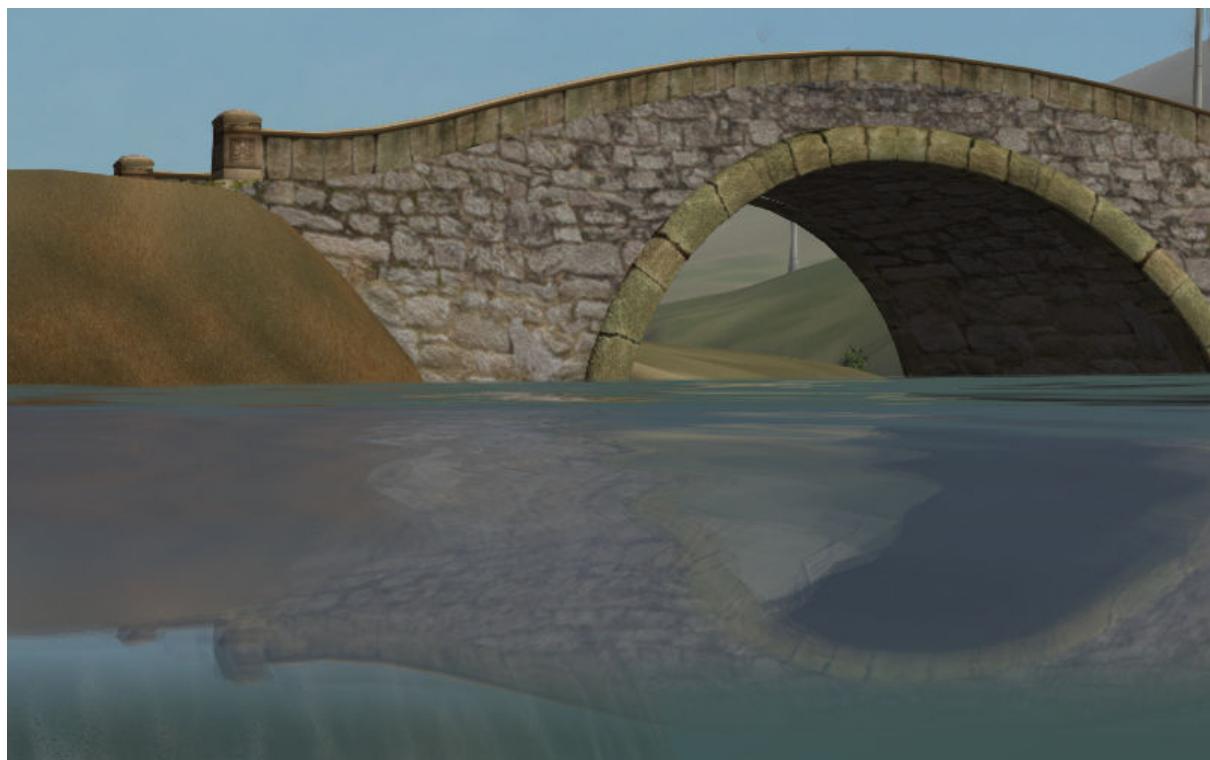
Visibility Settings (“fog”)

Underwater Fog > Color Fog color. The default value is (0.4, 0.6, 0.7).

Underwater Fog > Density Exponential factor which affects the density and visibility distance. The default value is 0.06.

The [god rays](#) effect settings are also applied.

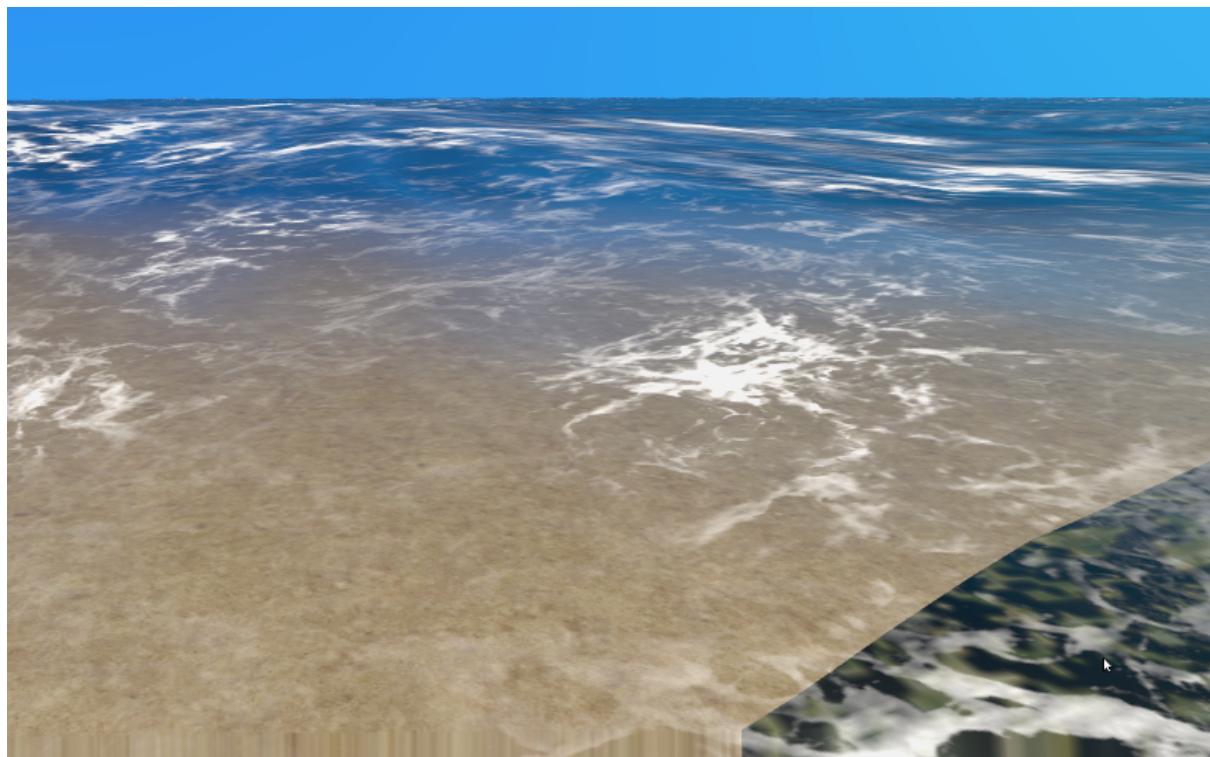
Note: The [Rendering Options](#) > Backface Culling option must be turned off for the correct water surface rendering.



Volumetric Waves

Activation

To enable procedural waves the Waves option must be turned on.



Setting up

Wave Height Wave height. The default value is 0.0.

Wave Length Wave length. The default value is 10.0.

Noise Dist Scale 0 Size of the first component of the open water waves.

Noise Dist Scale 1 Size of the second component of the open water waves.

Noise Dist Freq 0 Frequency of the first component of the open water waves.

Noise Dist Freq 1 Frequency of the second component of the open water waves.

Min Dir Shore Fac Minimum height decrease coefficient of the shore waves.

Dir Frequency Frequency of the rolling of the shore waves.

Noise Dir Scale Noise size for the shore waves.

Noise Dir Freq Noise frequency for the shore waves.

Min Dir Noise Fac Noise minimum for the shore waves.

Min Dist Fac Minimum coefficient of mixing for open water waves.

Horizontal Factor Coefficient that shows how much the shore waves are shifted in the shoreline direction.

Settings for Surface Generation

Generate Mesh Enable generated surface.

Cascades Number Number of cascades in the generated surface.

Subdivisions Generated mesh subdivisions.

Detailed Distance Maximum distance from camera to the last cascades edge.

Baking Shoreline Data to Texture

On the tools panel (hotkey “T”) under the Blend4Web tab open the Bake Shore Distance Map panel. Set the parameters: maximum distance to shore (Maximum Distance) and the resulting texture size (Texture Size). Select a landscape object (or multiple objects) first, and then - a water object. Click the Bake button.

Depending on the texture size and the number of vertices in the processed meshes the execution time of the script may vary from a fraction of a second up to several minutes. Make sure that the texture named ShoreDistance is created for the water mesh.

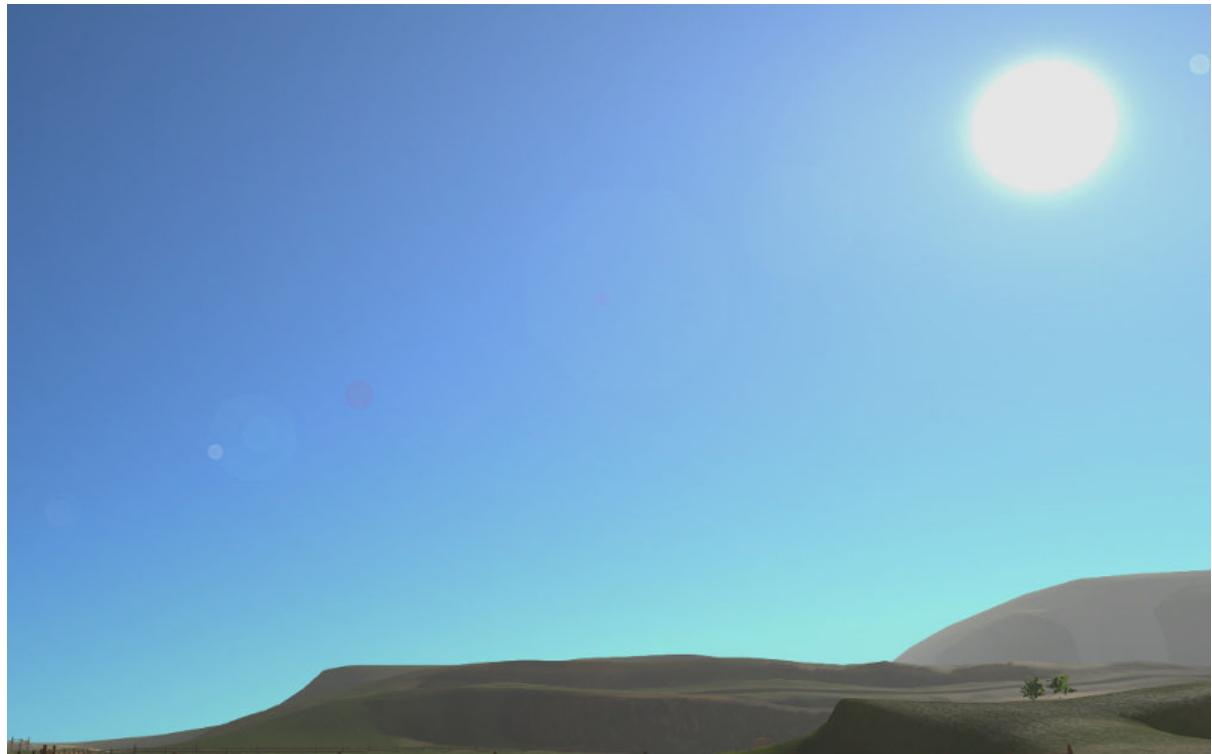
Upon script execution some system properties are saved in the water material. Therefore, the scene must be saved after the script has finished working.

Atmosphere

Scattering

Enable World > Render Sky, then activate Procedural Sky panel under the World tab. Please note, that if a [static skydome texture](#) is being used at the same time, it will be replaced.

Note: Also, a procedural sky texture can be used to imitate scattered [environment lighting](#) similar to the [static skydome texture](#). To do this, enable the Procedural Sky > Use as Environment Lighting and Environment Lighting > Sky Texture options. If the world texture for environment lighting already exists, it will be replaced.



Supported settings:

Procedural Sky > Sky Color Base sky color. The default value is (0.087, 0.255, 0.6) (blue).

Procedural Sky > Rayleigh Brightness Rayleigh scattering brightness (i.e. scattering on small particles). The default value is 3.3.

Procedural Sky > Mie Brightness Mie scattering brightness (i.e. scattering on large particles). The default value is 0.1.

Procedural Sky > Spot Brightness Sun spot brightness. The default value is 20.0.

Procedural Sky > Scatter Strength Light scattering factor. The default value is 0.2.

Procedural Sky > Rayleigh Strength Rayleigh scattering factor. The default value is 0.2.

Procedural Sky > Mie Strength Mie scattering factor. The default value is 0.006.

Procedural Sky > Rayleigh Collection Power Rayleigh scattering exponent. The default value is 0.35.

Procedural Sky > Mie Collection Power Mie scattering exponent. The default value is 0.5.

Procedural Sky > Mie Distribution Mie scattering distribution. The default value is 0.4.

Fog

The engine supports standard parameters located on the World > Mist panel. Besides this, overriding fog color is possible.

Mist > Use Custom Color Apply the user-defined fog color instead of World > Horizon color. Enabled by default.

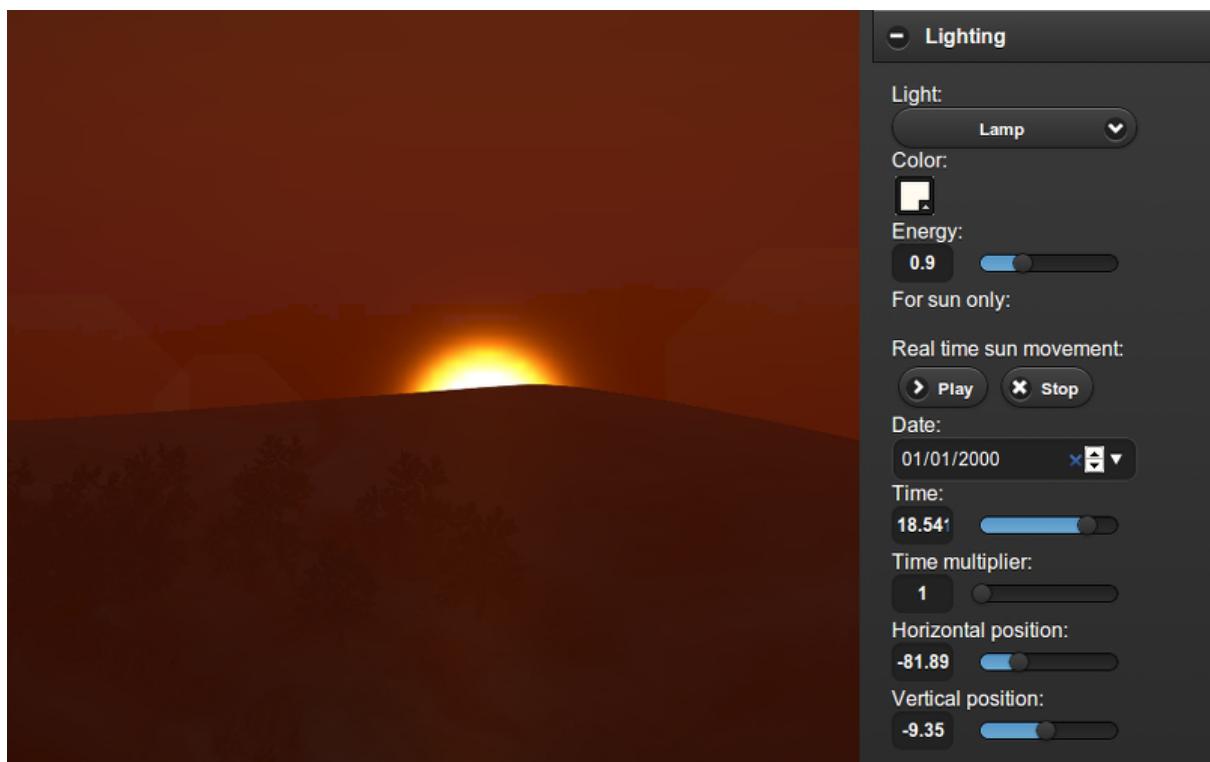
Mist > Color Fog color. The default value is (0.5, 0.5, 0.5) (gray).

When a dynamic skydome is used the fog color is defined by the sky color.

Time of Day

Enable the Dynamic Intensity options for the lamp.

Time of day can be set by applications via API. Particularly time of day can be set using the Lighting interface of the [Scene viewer](#).



Stars

Stars setup is described in the [Halo Materials](#) section.



Wind

Wind strength and direction affect:

- grass and tree leaves animation
- particle system dynamics
- water waves rolling frequency (at the moment only strength is taken into account)

Activation

Add a force field object of the Wind type.

Setting up

Direction Direction can be set by rotating the force field object.

Force Fields > Strength Wind strength. Located under the Physics tab. The default value is 1.0.

Animation of Grass and Tree Leaves

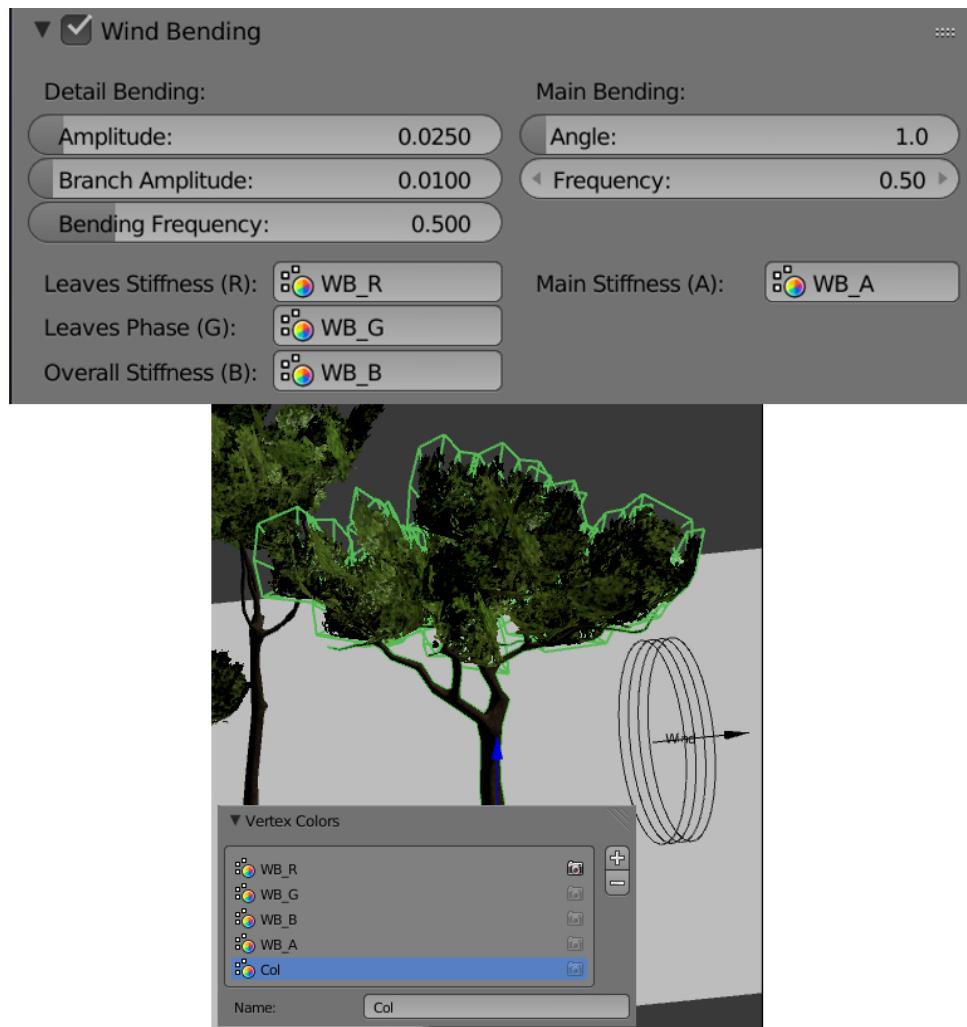
Authoring resources for grass rendering is described in the [Grass](#) section.

Activation

Enable the Wind Bending panel for the grass or tree object.

Setting up

The interface panel becomes visible after turning on the Wind Bending panel.



Main Bending > Angle Angle amplitude of the “main” deviation under the influence of wind (in degrees). The default value is 10.0.

Main Bending > Frequency Frequency of the “main” deviation under the influence of wind. The default value is 0.25.

Main Bending > Main Stiffness (A) Text field for specifying the name of the vertex color layer which contains the information about the stiffness of the “main” deviation. Can be left empty.

Detail Bending > Amplitude Angle amplitude of the “detail” deviation caused by the influence of wind (in degrees). The default value is 0.1.

Detail Bending > Branch Amplitude Angle amplitude of the branch deviation caused by the influence of wind (in degrees). The default value is 0.3.

Detail Bending > Bending Frequency Detail bending frequency. The default value is 1.0.

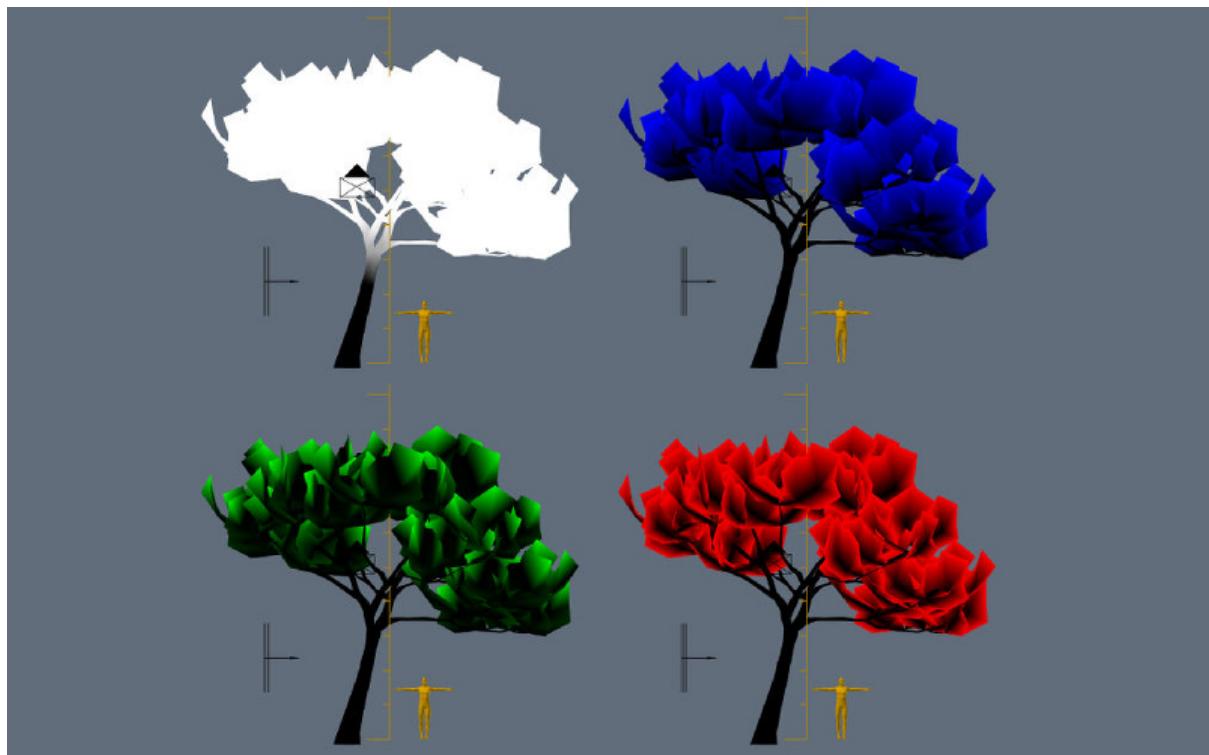
Detail Bending > Leaves Stiffness (R) Text field for specifying the name of the vertex color layer which contains the information about the stiffness of leaves. Can be left

empty.

Detail Bending > Leaves Phase (G) Text field for specifying the name of the vertex color layer which contains the information about the phase of leaves deviation. Can be left empty.

Detail Bending > Overall Stiffness (B) Text field for specifying the name of the vertex color layer which contains the information about the overall stiffness of leaves. Can be left empty.

Vertex color layers should be present in the mesh if their names are specified.



Color Management

Table of Contents

- Color Management
 - Gamma Overview
 - Human Vision and Monitors
 - Gamma Formula
 - Gamma in Node Materials
 - * Nodes for Coloring
 - * Nodes for Masking
 - * Normal Maps
 - * Summary Table
 - Alpha Compositing
 - * Overview
 - * Implementation
 - Color Correction
 - * Activation
 - * Additional settings

Gamma Overview

The essence of gamma correction is packing the image brightness channel into 8 bits of information.

Standard (non-HDR) images are always stored in non-linear color space where the darker components are encoded using more bits than the brighter ones. That means that a bigger RGB value corresponds to 0.5 of the real light intensity (a physical quantity called illuminance) - in the simplest case this value equals to $0.5^{(1/2.2)} = 0.73$.

Otherwise, 8 bit of information will not be enough to encode the light intensity. This will result in incorrect rendering of darker tones. For example, dark gradients will look stepped.

Therefore, web browsers, as well as many other programs for viewing and manipulating

images, work in non-linear space. However, 3D engines and renderers work in linear space, because it is the only correct way to represent light behavior in the real world. For example, the illuminance from two identical lamps exceeds the illuminance from one lamp exactly by two times.

Undoubtedly, 8 bit of information will not be enough in this case. This can be clearly seen from the table in which approximate illuminance values for some real light sources are shown.

Description	Illuminance, lux
Summer noon	17 000
Winter noon	5 000
Dull day	1 000
In a light room	100
Full moon by night	0.2
Moonless light	0.001

When the Color Management > Display Device > sRGB option is enabled for a scene, Blender works in a linear space. Material colors and lamp settings correspond to physical values. For texture images (except normal maps) it is required to select the Image > Input Color Space > sRGB option. In this case an automatic image unpacking (sRGB -> Linear) is performed at the time of rendering.

Human Vision and Monitors

While the human vision is non-linear (a human recognizes the darker light tints better than the brighter ones), the light coming into the eye still obeys the physical laws (see the lamps example).

In CRT monitors the brightness is dependent non-linearly upon the electric voltage applied to the monitor's input (the voltage itself is determined by the color channel value in the video memory). LCD monitors mimic the same characteristics. Nevertheless, the light emitted by such monitors obeys the physical laws. For example the addition of a second light source to a virtual scene should cause the brightness to double (in the perfect case).

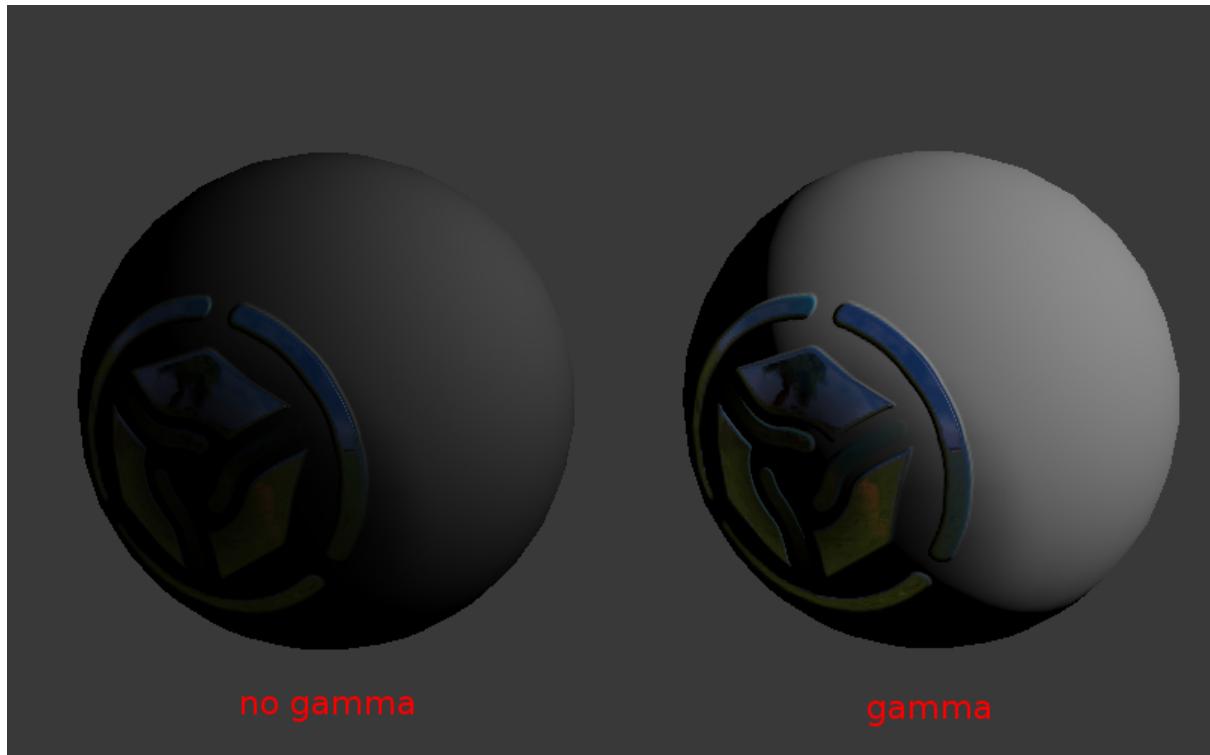
Therefore, the perception characteristics of the human eye are the reason due to which it is possible to pack color channels. At the same time, the technical characteristics of monitors have a secondary significance for gamma correction.

Gamma Formula

Used in the following simplified formula:

$$V_{\text{out}} = V_{\text{in}}^{\gamma}$$

$\gamma < 1$ - packing gamma, $\gamma > 1$ - unpacking gamma. In the simplest case 1/2.2 and 2.2 values are used respectively. Hereinafter the “packing” (Linear \rightarrow sRGB) and “unpacking” (sRGB \rightarrow Linear) terms are used instead of “gamma correction”.



Gamma in Node Materials

Nodes for Coloring

Unpacking (sRGB \rightarrow Linear) is required when textures and vertex colors are used for coloring (not for masking). The texture node and Vertex Color output of Geometry node implement unpacking automatically.

Note that the alpha channel of a texture node is not corrected. Its values are in the linear space.

Nodes for Masking

Textures and vertex colors can be used as masks i.e. input data for some mathematical operations. In such a case the packing operation is required.

Keep in mind that a texture node and Vertex Color output of Geometry node implement unpacking automatically. This results in necessity of the additional transformation back to the non-linear space, for which the LINEAR_TO_SRGB or GAMMA node with $\gamma = 1/2.2$ is used.

Note: LINEAR_TO_SRGB and SRGB_TO_LINEAR nodes are deprecated. GAMMA node should be used instead.

Normal Maps

No transformations are performed for normal maps.

Summary Table

Use case	Correction
Texture for coloring	implemented automatically in the texture node (the alpha channel is not corrected)
Texture for masking	GAMMA with $\gamma = 0.45$
Vertex color for coloring	implemented automatically
Vertex color for masking	GAMMA with $\gamma = 0.45$
Normal map	not required

Alpha Compositing

Overview

Physically correct alpha compositing is performed according to the formula [source <<http://wwwnew.cs.princeton.edu/courses/archive/spring05/cos426/papers.smith95a.pdf>>‘_]:

$$C_o = C_a \alpha_a + C_b \alpha_b (1 - \alpha_a).$$

This formula differs from the classic mix operation (aka convex combination) because it has the α_b multiplier in the second summand. Therefore, not only the α_a value of the source pixel should be known for alpha compositing, but also the α_b value of the pixel over which the rendering is performed.

In case of preliminary multiplication of the α values by the color channels (so called premultiplied alpha) the formula becomes as following:

$$C_o = C_a + C_b (1 - \alpha_a).$$

The last formula is used also to calculate the resulting α_o value:

$$\alpha_o = \alpha_a + \alpha_b (1 - \alpha_a).$$

Preliminary multiplication of the color channels by the α values allows to save two multiplication operations. The more significant thing is that the derived formula can be used

repeatedly without the need to divide the C_o color by the $\alpha o \alpha o$ value on each consequent iteration.

Implementation

The blending function used in Blend4Web is the following:

```
gl.blendFunc(gl.ONE, gl.ONE_MINUS_SRC_ALPHA);
```

WebGL context initialization is performed using the premultipliedAlpha = true parameter (that is the default value). Also multiplication of all the color channels by the α value is performed on the output of the shaders.

Color Correction



Activation

Activate the Color Correction panel under the Render tab.

Additional settings

Brightness The default value is 0.0.

Contrast The default value is 0.0.

Exposure The default value is 1.0.

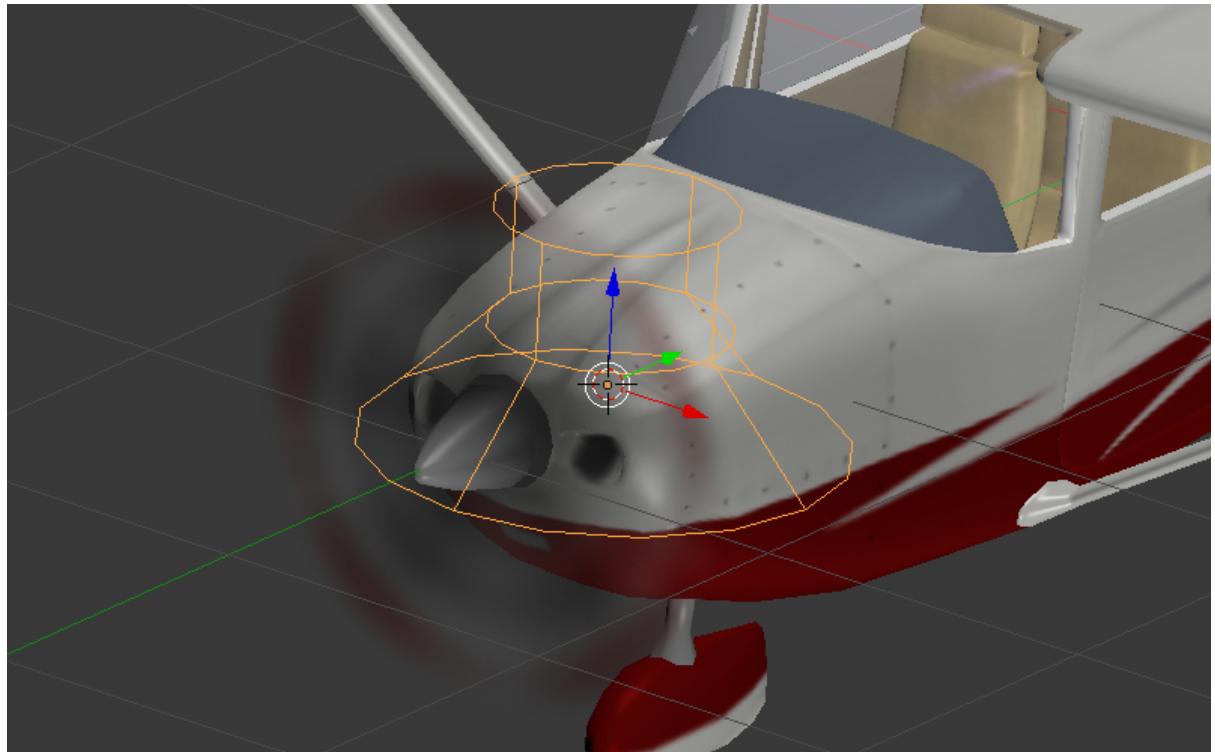
Saturation The default value is 1.0.

Audio

Table of Contents

- [Audio](#)
 - [Audio Source Settings](#)
 - [Processing and Decoding](#)
 - * [Supported formats \(containers\):](#)

Audio sources are created in Blender. The standard Speaker object is used.



Audio Source Settings

Speaker parameters can be set up on the Properties panel under the Object Data tab. The engine supports all the standard Blender sound parameters and some engine-specific settings.

Speaker Behavior:

The behavior of the audio source.

Positional — high-quality sound with spatial positioning and directivity (conicity). The Web Audio API is used for sound rendering. Playback performance of such sounds is the least and so use them only for short samples.

Background Sound — high-quality omnidirectional sound without spatial positioning. The Web Audio API is used for sound rendering. It is more performant but is not effective for music.

Background Music — used for music playback. It has maximum performance due to the use of the Audio HTML tag, but has minimum flexibility.

The following options are available on the Sound panel:

Volume

Speaker volume

Random Volume

Additional volume randomization. The resulting value is calculated as for the delay.

Pitch

Sound playback velocity.

Random Pitch

Additional randomization of the sound playback speed. The resulting value is calculated as for the delay.

Fade-In

Fade-in time interval.

Fade-Out

Fade-out time interval.

Disable Doppler

Ignore source's frequency shift upon its moving.

Cyclic Play

Loop the sound playback.

Delay

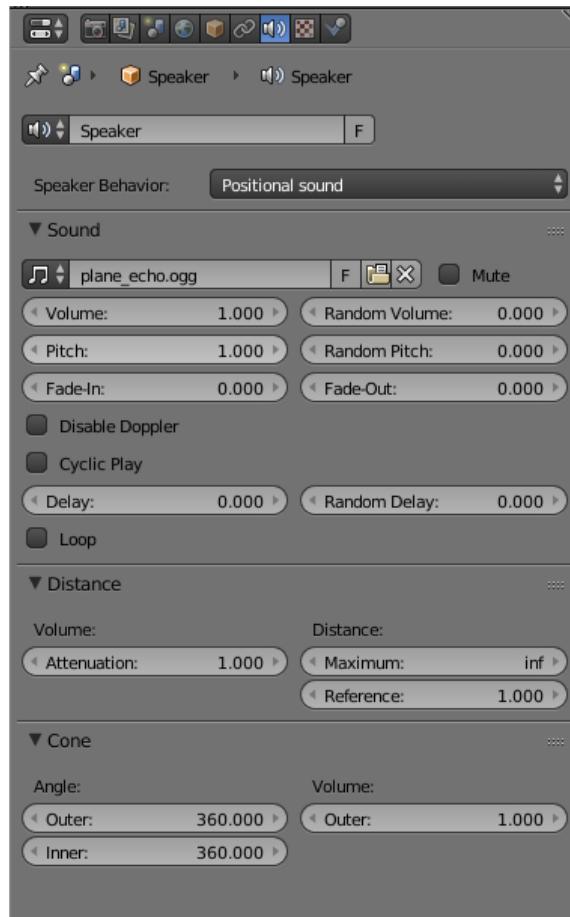
Delay before sound playback starts.

Random Delay

Additional delay randomization. The resulting value is calculated according to the formula $Delay_{result} = Delay + Delay_{random} * Random_{[0-1]}$.

Loop

Loop the sound playback. Contrary to the Cyclic play option it guarantees a zero delay upon repeat. The option is available only for sound sources with Positional or Background Sound behavior.



Processing and Decoding

Supported formats (containers):

- ogg, Vorbis codec (Chrome, Firefox)
- mp3 (Chrome, Safari)

- mp4, AAC codec (Chrome, Safari)

It is recommended to use Ogg as it is an open standard, is widespread in browsers and provides good sound quality. The optimal format in respect to the quality and compatibility is 48kHz/16bit. Single-channel sound (mono) is used to store shot samples while two-channel sound (stereo) is used for music playback.

Converting resources between different formats is described in the [corresponding section](#).

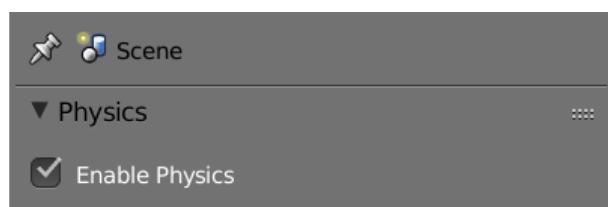
Physics

Table of Contents

- Physics
 - Preparing for Use
 - Static Physics Type
 - Dynamic Physics Type
 - Constraints
 - Wheeled Vehicles
 - * Chassis Settings
 - Floating Objects
 - * Floating Object Settings
 - Floating Vehicles aka Watercrafts
 - * Watercraft Settings
 - Use in Applications

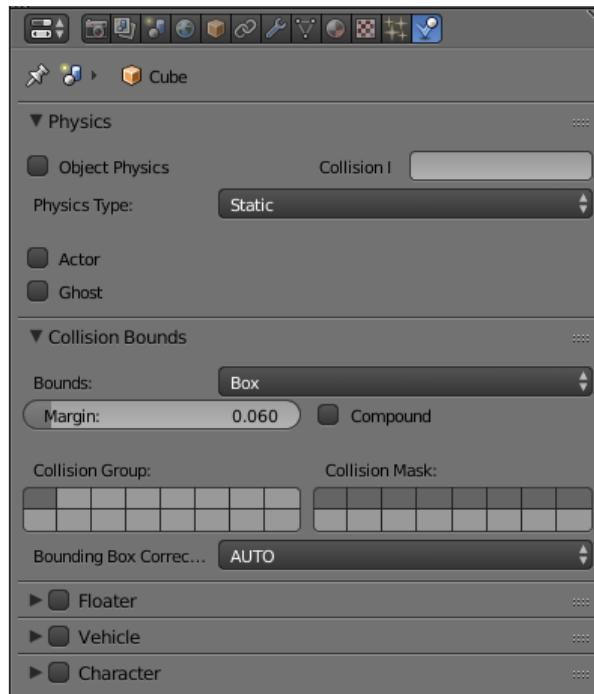
Preparing for Use

In order to enable physics on the scene, please use the Enable Physics checkbox in the Physics panel under the scene tab in Blender.

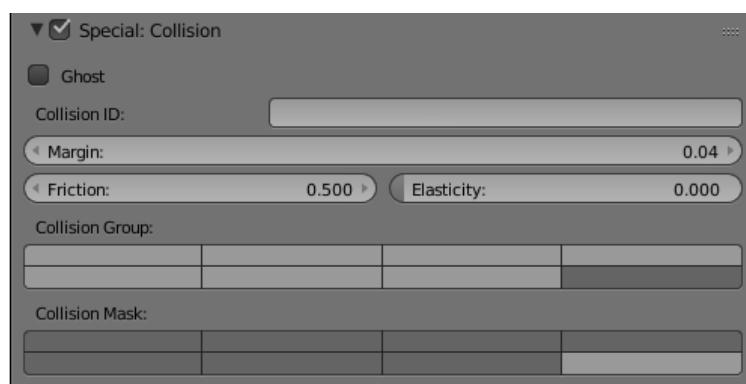


Static Physics Type

Can be used to limit the movement of other objects, for example to detect collisions with a landscape, walls and so on. In the physics settings of such an object the Static value (set by default) should be selected for the Physics Type option.

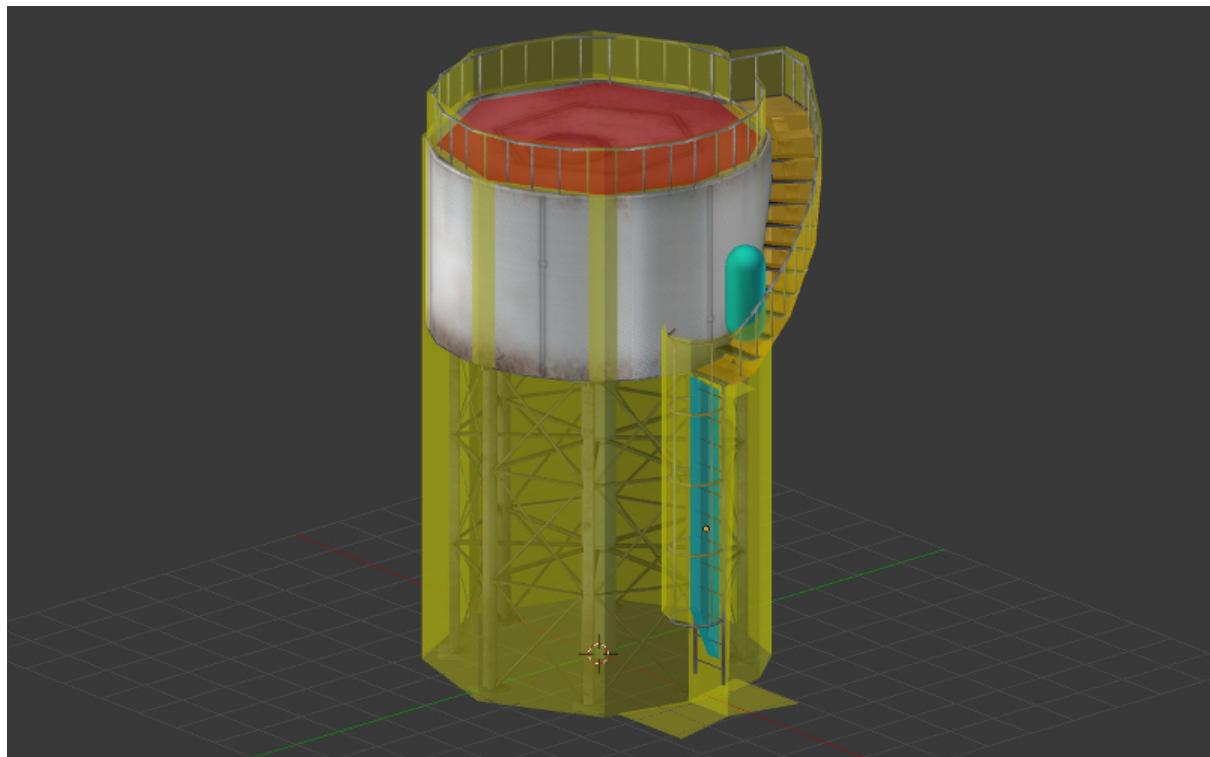


One or multiple physics materials can be assigned to a mesh. Under the Material tab the Special: Collision panel must be activated.



The Ghost option excludes the material from physical interactions but still notifies the

application about the contact with it. An example - detecting that the character is located on a vertical ladder.



The Collision ID field is intended for detecting collisions with specific materials and can be left empty. An example of Collision ID usage is detecting the landscape surface a character is located on - grass, sand, wooden coating and so on.

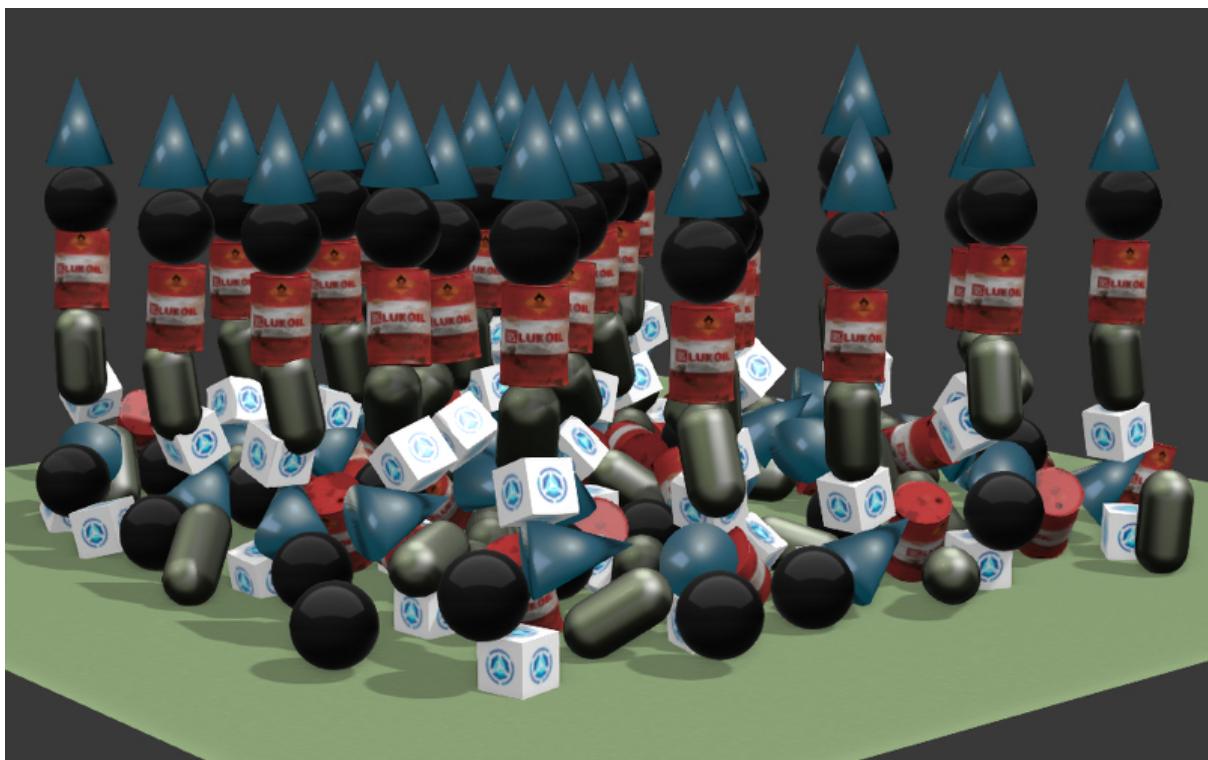
The Margin field allows to customize the width of the zone where mesh reacts on collisions. This option improves physical collisions simulation stability.

Also, there are material physics settings in this panel. The following settings are supported: Friction, Elasticity.

The Collision Group field corresponds to the physics group which the material belongs to. The Collision Mask field defines all physics groups with which this material will interact.

Dynamic Physics Type

Intended for rigid body movement simulation.

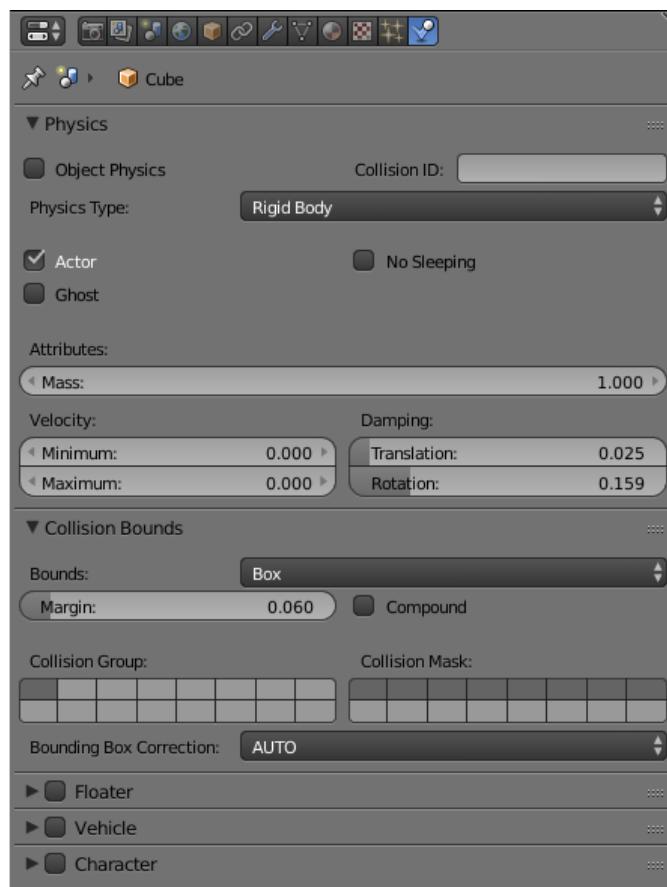


The Object Physics checkbox must be enabled under the object's Physics panel. The Collision ID field is intended for detecting collisions with a specific object (for example, for detecting proximity of a FPS character to different items) and can be left empty.

In the physics settings of such an object the Rigid Body (with rotations) or Dynamic (without rotations) values can be selected for the Physics Type option. In the Collision Bounds settings the collider type can be selected - the supported types are: Box, Capsule, Sphere, Cylinder, Cone. Also, the following physics parameters can be set: Mass, Damping - for Translation and Rotation.

The Collision Group field corresponds to the physics group which the object belongs to.

The Collision Mask field defines all physics groups with which this object will interact.

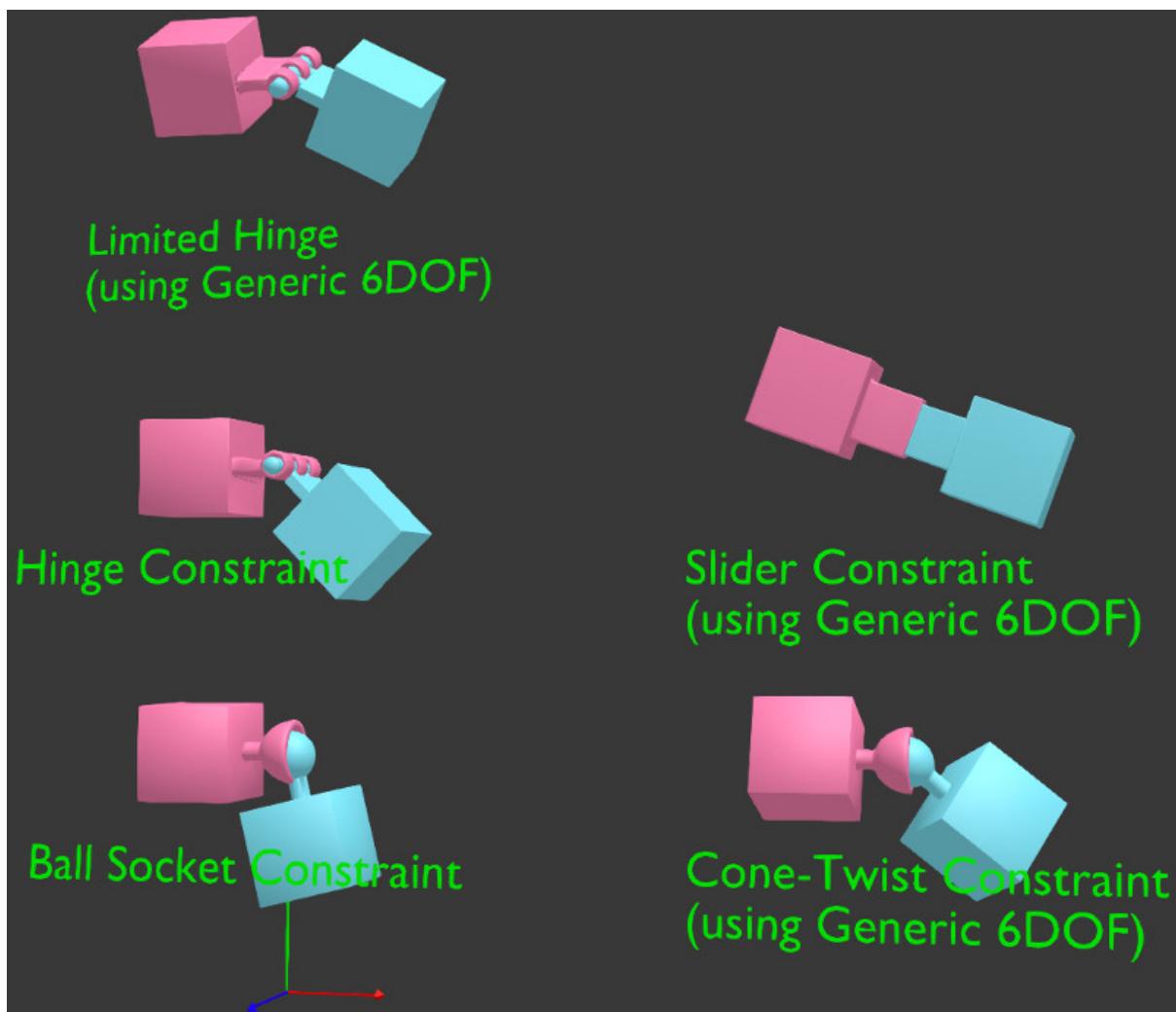


Friction and Elasticity are supported for the material of such an object. When multiple materials are used on a single mesh, the physics parameters are taken from the first of them.

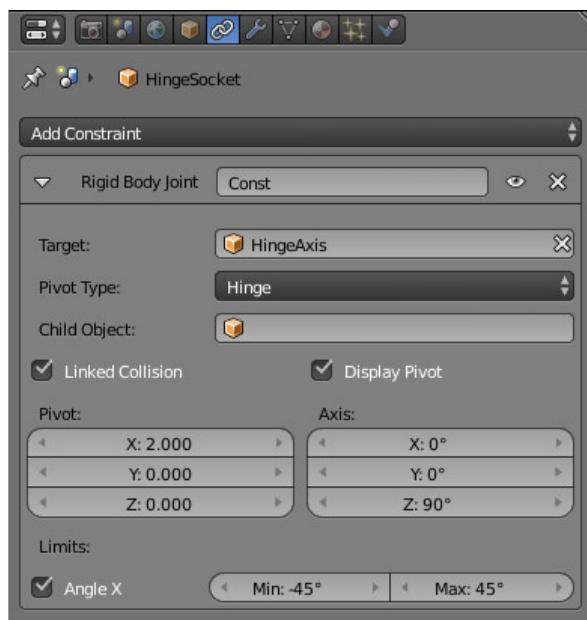
For the camera object the Physics Type = Dynamic parameter must be used, and the Object Physics checkbox must be enabled.

Constraints

Physical constraints are used for limiting the objects' degrees of freedom.

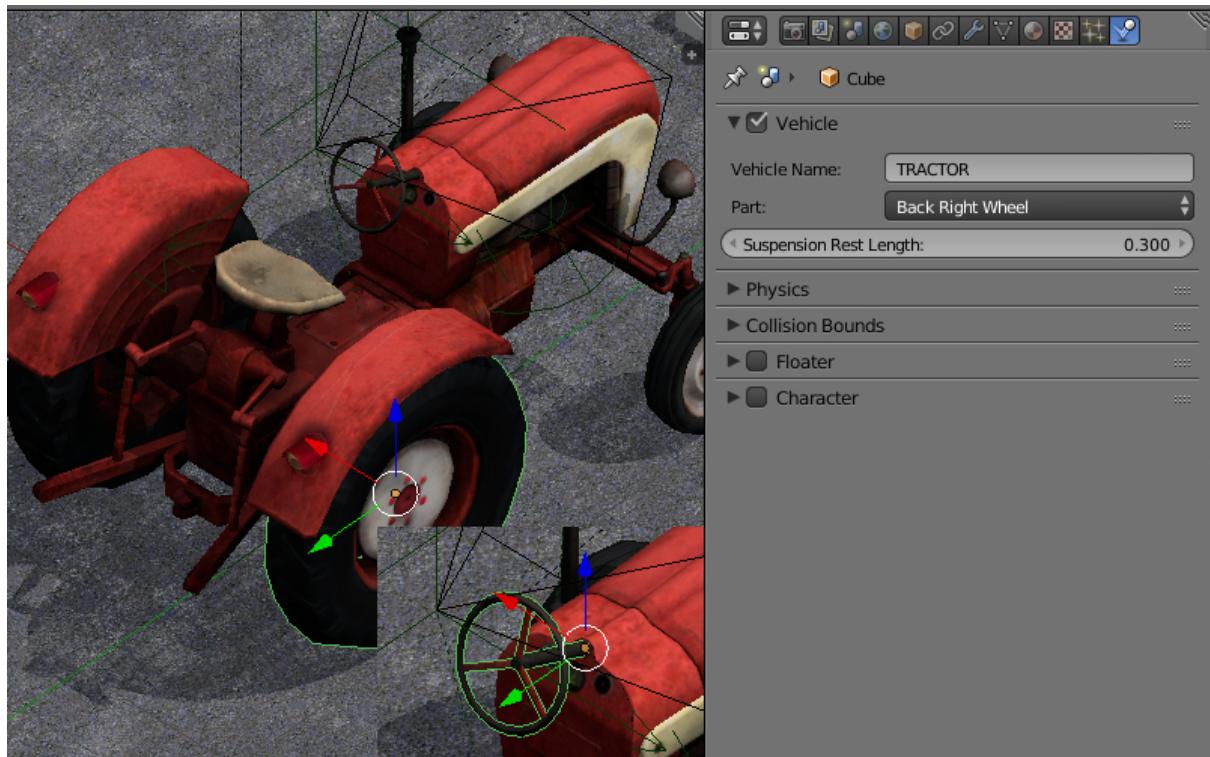


Adding a physical constraint (Rigid Body Joint) to the object can be performed on the Object Constraints panel. The supported types (Pivot Type) are: Ball, Hinge, Cone Twist, Generic 6 DoF. A physical constraint can be added to one of the two interacting objects, while the other object acts as a Target. Both objects can have a static and/or dynamic physics type. In constraints (except Ball) the translation and rotation limits can be set up.



Wheeled Vehicles

The model of a vehicle must consist of 6 separate objects - a chassis, 4 wheels and a steering wheel. The chassis' mesh center should correspond to the mass center. The centers of the wheels' and the steering wheel's meshes should be located on the rotation axes. The steering wheel should be oriented in the local space of coordinates - X - the rotation axis, Y - to the right and Z - upwards. The object can have any names.



For all 6 objects: select the Part, specify the same id in the Vehicle Name field, select the right object type - Chassis, Steering Wheel, Back Right Wheel and so on. The Suspension Rest Length setting is also available for the wheels.

It is necessary to specify a realistic mass for the chassis (because the default value is only 1 kg). To do this go to the physics settings, choose the Rigid Body value for the Physics Type option and specify the required value (for example, 1000 kg) in the Mass field.

Chassis Settings

Force Max Maximum driving force of the vehicle.

Brake Max Maximum braking coefficient.

Suspension Compression Damping coefficient for suspension stretching.

Suspension Stiffness Suspension stiffness coefficient.

Suspension Damping Suspension damping coefficient.

Wheel Friction Friction constant between the wheels and the surface. It should be around 0.8 for realistic vehicles. But it can be increased significantly to achieve a better control (1000 and more).

Roll Influence Decreases the wheels' torque decreasing the probability of the vehicle overturning (0 - no torque, 1 - real physics behavior).

Max Suspension Travel Cm Maximum suspension travel in centimeters.

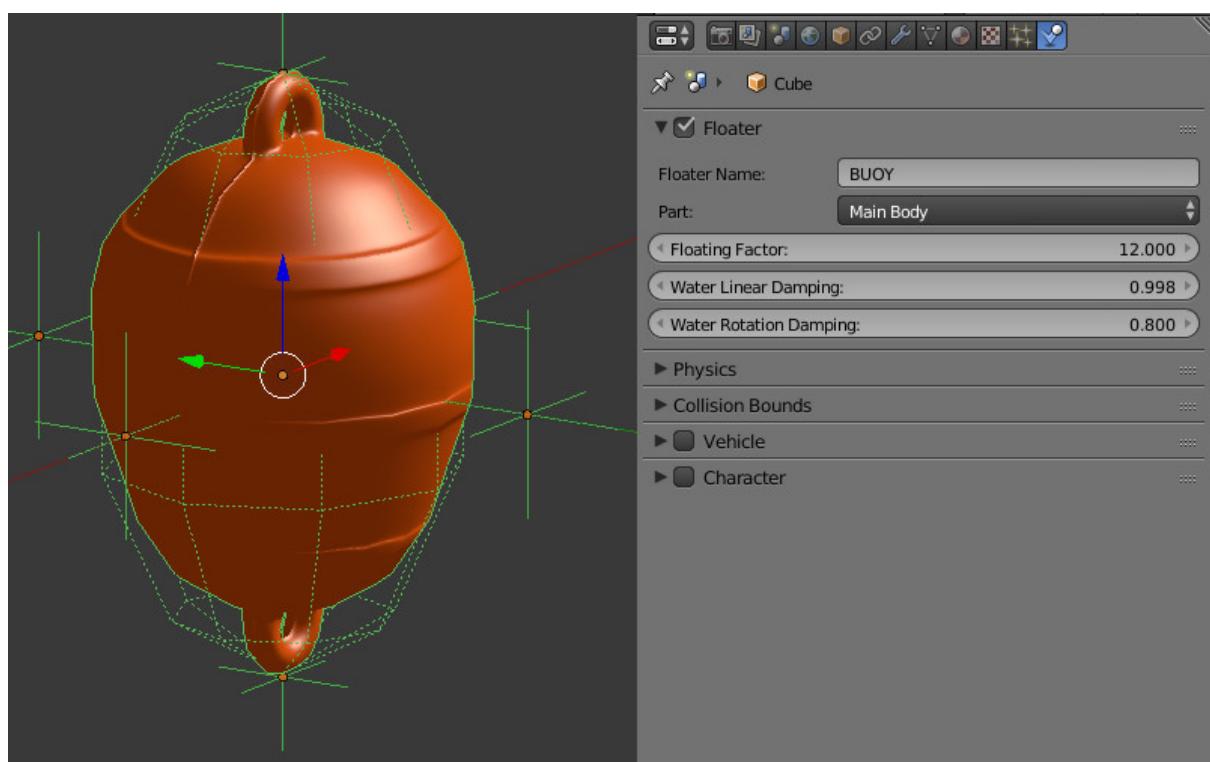
For the Steering Wheel it is necessary to specify the maximum steering angle (Steering Max) and the ratio between the turn of the steering wheel and the turn of the wheels (Steering Ratio). The maximum steering angle value is specified in revolutions. A single revolution equals to 360 degrees. Therefore, if Steering Max is equal to one and Steering Ratio is equal to 10, the maximum turn of the steering wheel will be 360 degrees and the maximum turn of the front wheels will be 36 degrees.

On this stage you can export and load the scene into the engine. We recommend to create a road surface with a physics material. To choose the controlled object press the Q key in the Viewer and select the chassis. Use the W, A, S, D keys as controls.

We can additionally tweak the Damping of Translation and Rotation. This will influence the speed and inertia of the vehicle.

The friction and elasticity of the road surface material do not influence the vehicle's behavior.

Floating Objects



In order for the object to float on the water surface (an object with the Water material), it is necessary to enable the Floater panel. There are two types of floating objects:

Main Body - the floating object itself and Bob - an auxiliary bob-object onto which the buoyancy will be acting. A floating object can have an unlimited number of Bob objects. This can be both meshes or Empty objects.

All objects that are part of the same floating object must have the same name in the Floater Name field.

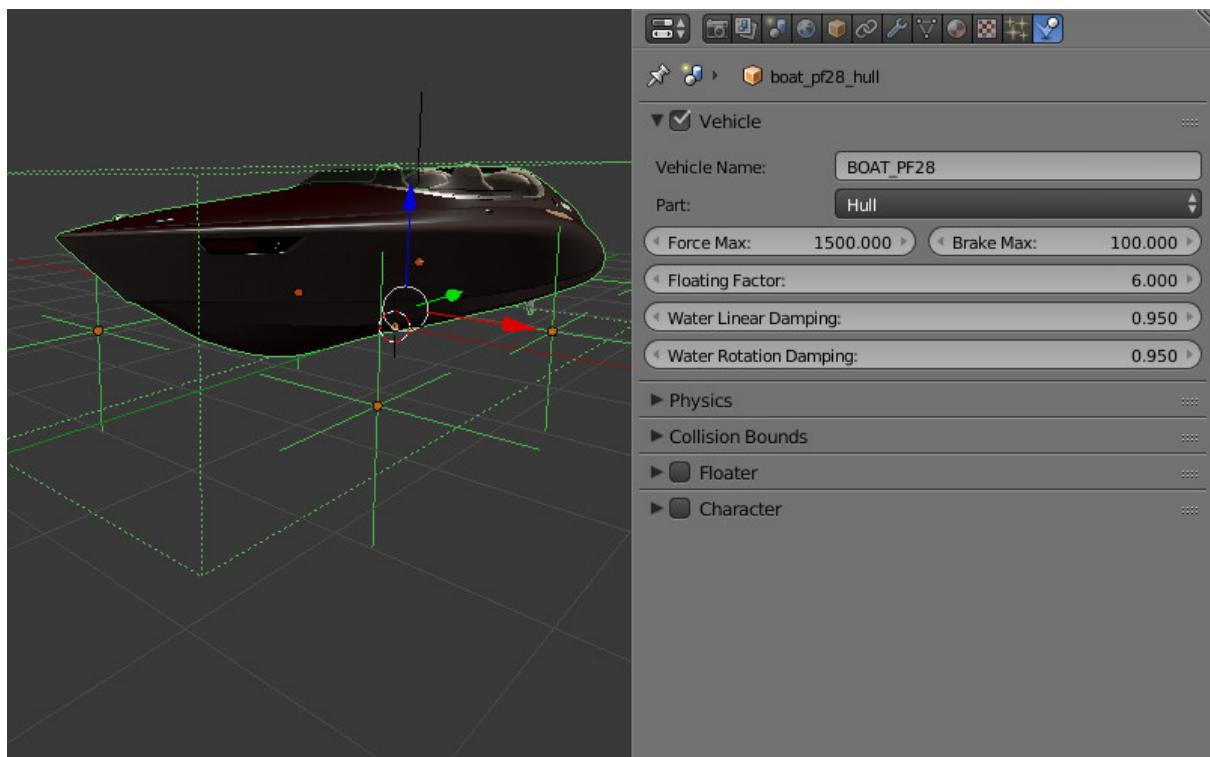
Floating Object Settings

Floating Factor Buoyancy coefficient.

Water Linear Damping Linear velocity damping when the object is on the water surface (or under water). When the object is not in water, the physics settings are used.

Water Rotation Damping Rotation damping when the object is on the water surface (or under water). When the object is not in water, the physics settings are used.

Floating Vehicles aka Watercrafts



Watercrafts use some parameters from the Vehicle settings and all the settings which are similar to Floater setting. It is necessary to set the Part type Hull on the main object. Similar to a floating object a watercraft requires auxiliary Bob objects.

Watercraft Settings

Force Max Maximum driving force of the vehicle.

Brake Max Maximum braking coefficient.

Floating Factor Buoyancy coefficient.

Water Linear Damping Linear velocity damping when the object is on the water surface (or under water). When the object is not in water, the physics settings are used.

Water Rotation Damping Rotation damping when the object is on the water surface (or under water). When the object is not in water, the physics settings are used.

Use in Applications

The physics system is implemented in the `uranium.js` module and loaded separately from the engine's main code. The `uranium.js` module itself is a modification of the [Bullet](#) physics engine, which is ported to work in browsers. In order to activate the physics system, it is enough to put the `uranium.js` and `uranium.js.mem` files in the same directory as the source code of the application.

Another way is to explicitly specify the loading path of the `uranium.js` module by using the following API method:

```
m_config.set("physics_uraniun_path", ".../uranium.js");
```

Note: When applications are developed [within the SDK](#), the path to the physics engine is detected automatically.

If your application does not use physics, we recommend you to turn off the `Enable Physics` flag in the `Physics` panel under the `Scene` tab in Blender. It is also possible to forcibly disable loading of the `uranium.js` module by calling the following method before initialization of the engine:

```
m_config.set("physics_enabled", false);
```

For Application Developers

Table of Contents

- For Application Developers
 - Project Management
 - Advanced Project Management
 - * Dependencies
 - * Projects List
 - * Project Structure
 - * .b4w_project Configuration File
 - * Advanced Project Management
 - * Developing multiple apps inside a project
 - * Building Projects
 - * Automatic Re-export
 - * Resource Conversion
 - * Deploying Projects
 - * Remove Project
 - * Upgrading Apps for New SDK Versions
 - * Path to Loaded Application Assets
 - Application Programming
 - * “Hello World!” application
 - * Loading Scenes in Apps
 - * Module System
 - * Creating Apps Quickly
 - Resource Conversion
 - * Dependencies
 - * Data Format
 - Code Examples
 - Event-Driven Model
 - * Sensors
 - * Example
 - SDK File Structure
 - Loading Local Resources
 - Quality Profiles
 - Non-FullScreen Web Apps

Project Management

You can manage your projects in two ways: by using `project.py`, which is a simple-to-use and well documented CLI utility, or by using a dedicated web application with GUI working on the local development server. The server does not require any additional setup and starts automatically with Blender. The index page of the Project Manager can be opened by clicking on the Project Manager button in Blender.

Working with the Project Manager is described in a dedicated chapter.

Advanced Project Management

Advanced project management is used by experienced developers which require more flexibility and need to automate process of project development.

For advanced project management use the `project.py` script and manually edit `.b4w_project` configuration files.

Dependencies

The project management system works in all operating systems. However, some operations can require installing additional dependencies. In order to check whether all dependencies are met, use the following command:

```
./project.py check_deps
```

For MS Windows users:

```
python project.py check_deps
```

Projects List

```
python3 project.py -p myproject list
```

Prints list of projects in the SDK.

Project Structure

A typical app developed using the project manager looks as follows:

```
blend4web/
  apps_dev/
    myproject/
      project.py
      .b4w_project
      myproject.js
      myproject.css
      myproject_dev.html
  blender/
    myproject/
      myproject.blend
  deploy/
    apps/
      myproject/
        myproject.js
        myproject.css
        myproject.html
```

```
assets/
  myproject/
    myproject.json
    myproject.bin
```

This app consists of 4 different directories.

1. apps_dev/myproject. Contains source files of project's apps.
2. blender/myproject. Contains source files of project's scenes.
3. deploy/assets/myproject. Contains exported files of project's scenes.
4. deploy/apps/myproject. Contains exported files of project's scenes.

Additionally, the deploy command can create yet another directory, but it's usually placed outside of the SDK and its name and path depend on directory structure on the target server.

.b4w_project Configuration File

If you did not use any arguments upon executing the project.py script, then they will be taken from the configuration file.

```
[info]
author =
name =
title =
icon =

[paths]
assets_dirs =
blend_dirs =
blender_exec =
build_dir =
deploy_dir =

[compile]
apps =
css_ignore =
engine_type = external
js_ignore =
optimization = simple
use_physics =
use_smaa_textures =
version =

[deploy]
assets_path_dest =
assets_path_prefix =
remove_exist_ext_dir =
```

Advanced Project Management

```
./project.py init myproject
```

This command will create a project with the specified name in the current directory. By default the project directory will only contain a config file.

Available parameters:

- -A | --copy-app-templates (optional) create standard app templates in the project directory (<my_project_name>_dev.html, <my_project_name>.js, <my_project_name>.css).
- -B | --bundle (optional) all project files will be placed in a single directory.
- -C | --author (optional) write an author's or a company's name in the config file.
- -o | --optimization (optional) write the script optimization level in the config file.
- -P | --copy-project-script (optional) create a copy of the project.py script in the project directory.
- -S | --copy-scene-templates (optional) create standard scene templates in the directories deploy/assets/<my_project_name> and blender/<my_project_name> (<my_project_name>.json/.bin and <my_project_name>.blend correspondingly).
- -T | --title" (optional) write a title in the config file. Upon building, it will be used inside the <title> HTML element.
- -t | --engine-type (optional) write an engine type in the config file.

Example:

```
./project.py init -AS -C Blend4Web -o simple -T MyProject -t external myproject
```

This command will create a directory named myproject, inside which the following files will be placed: myproject.js, myproject.css, myproject_dev.html and .b4w_project.

The .b4w_project file will look like:

```
[info]
author = Blend4Web
name = myproject
title = MyProject

[paths]
assets_dirs = deploy/assets/myproject;
blend_dirs = blender/myproject;
blender_exec = blender
build_dir = deploy/apps/myproject
deploy_dir =

[compile]
apps =
```

```
css_ignore =
engine_type = external
js_ignore =
optimization = simple
use_physics =
use_smaa_textures =
version =

[deploy]
assets_path_prefix =
remove_exist_ext_dir =
```

Developing multiple apps inside a project

A project can contain multiple apps. This can be provided by listing the corresponding HTML files in the config file separated with semicolon:

```
...
[compile]
apps = myapp1;myapp2;
...
```

Building Projects

```
python3 project.py -p myproject compile
```

Builds a project in the deploy/apps/myproject directory.

Note: For script operation it's required to install java and set the PATH system variable.

Available parameters:

- "-a | --app" (optional) specify an HTML file, relative to which the project app will be built.
- "-c | --css-ignore" (optional) add CSS styles to exceptions in order to not compile them.
- "-j | --js-ignore" (optional) add scripts to exceptions in order to not compile them.
- "-o | --optimization" (optional) specify the optimization level for JavaScript files: whitespace, simple (by default) or advanced.
- "-t | --engine-type" (optional) define a compilation type for an app. Four variants are available: external (by default), copy, compile, update.
- "-v | --version" add version to paths of scripts and styles.

Compiler Requirements

- In the root of the directory the single html file must be stored if -a option is disabled
- Scripts and styles can be stored in the app's root and in the subfolders

Automatic Re-export

```
python3 project.py -p myproject reexport
```

This command will re-export blend files in JSON and HTML formats.

Available parameters:

- "-b | --blender-exec" path to the blender executable.
- "-s | --assets" specify directory with scene assets.

Resource Conversion

```
python3 project.py -p myproject convert_resources
```

Converts external resources (textures, audio and video files) into alternative formats to ensure cross-browser and cross-platform performance.

Available parameters:

- "-s | --assets" specify directory with scene assets.

Converting of resources is described in detail in the corresponding section.

Deploying Projects

```
python3 project.py -p myproject deploy DIRECTORY
```

Save a project to an external directory together with all dependencies.

Available parameters:

- "-e | --assets-dest" destination assets directory ("assets" by default).
- "-E | --assets-prefix" assets URL prefix ("assets" by default).
- "-o | --override" remove directory if it exists.
- "-s | --assets" override project's assets directory(s).
- "-t | --engine-type" override project's engine type config.

Remove Project

```
python3 project.py -p myproject remove
```

Removes a project. Removed directories are retrieved from project configuration file.

Upgrading Apps for New SDK Versions

While upgrading for new SDK versions often two problems arise:

1. Modules of the new and old versions of the engine do not match.
2. Old and new engine API do not match.

In order to update the list of modules imported in developer version of application go to project source directory apps_dev/my_project and execute module list generator script:

```
python3 ../../scripts/mod_list.py
```

For MS Windows users:

```
python ..\..\scripts\mod_list.py
```

Note: To run the scripts the Python 3.x needs to be installed in your system.

The console will print the list of modules - copy them and paste into the main HTML file:

```
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1">
  <script type="text/javascript" src="../../src/b4w.js"></script>
  <script type="text/javascript" src="../../src/anchors.js"></script>
  <script type="text/javascript" src="../../src/animation.js"></script>
  <script type="text/javascript" src="../../src/assets.js"></script>
  <script type="text/javascript" src="../../src/batch.js"></script>
  <script type="text/javascript" src="../../src/boundings.js"></script>
  <script type="text/javascript" src="../../src/camera.js"></script>
  ...
</head>
```

To eliminate API incompatibilities you may require refactoring of your app. All changes are described in [release notes](#).

Path to Loaded Application Assets

To load .json-files you should use `get_std_assets_path()` method from the config.js module:

```
m_data.load(m_config.get_std_assets_path() + "example/example.json", load_cb);
```

After building the finished app, the paths to assets will change. Thus, using `get_std_assets_path()` will allow you to avoid problems with incorrect paths.

Application Programming

“Hello World!” application

The simplest Blend4Web app may look like this:

```
<!DOCTYPE html>
<html>
<head>
<script src="b4w.min.js"></script>
<script>
function hello() {
    var m_version = b4w.require("version");
    document.body.innerHTML = "Hello, Blend4Web " + m_version.version() + "!";
}
</script>
</head>

<body onload="hello()"></body>

</html>
```

This app prints a message and the engine’s version in the browser window. Let’s look at this example in detail. The engine library is embedded with the `<script src="...">` element. Then, the app waits for the page to load and prints the current version in the browser window. In this example, `version` is the only used module which has a function with the same name - `version()`. A more detailed info about the usage of engine’s modules and functions can be found in the [API documentation](#).

The compiled engine file `b4w.min.js` can be copied from the SDK’s `deploy/apps/common` directory and placed in the same directory as the HTML file.

Loading Scenes in Apps

To load a 3D scene you need:

1. Place a `<canvas>` element on a page for rendering.
2. Call the `m_main.init()` function with the created element id to initialize the WebGL context after the page is loaded.
3. Call the `m_data.load()` function to load a 3D scene.

```
<!DOCTYPE html>
<html>
<head>
<script src="b4w.min.js"></script>
<script>
function hello() {
    var m_main = b4w.require("main");
```

```

var m_data = b4w.require("data");

var canvas_elem = document.getElementById("canvas_id");
m_main.init(canvas_elem);
m_data.load("some_scene.json");
}

</script>
</head>

<body onload="hello()"><canvas id="canvas_id"></canvas></body>

</html>

```

Note: Note that a real app should include error checking, setting up the engine before initializing and also a basic system for interacting with the user.

Module System

While the engine gives an app programmer an API in the scale of dozens of modules, it occupies a single b4w namespace. To call a module's method import it first with the b4w.require() function.

It is possible to register external modules if their names do not collide with already existing modules. A module can be registered with a b4w.register() call. Check if a module with some name already exists with a b4w.module_check() call.

Example:

```

// check if module exists
if (b4w.module_check("my_module"))
    throw "Failed to register module: my_module";

// register my_module
b4w.register("my_module", function(exports, require) {

    // import module "version"
    var m_version = require("version");

    // export print_build_date() from module "my_module"
    exports.print_build_date = function() {
        // exec function date() from module "version"
        console.log("Engine build date: " + m_version.date());
    }
});

// import module "my_module"
var m_my_module = b4w.require("my_module");

```

```
// exec function print_build_date() from module "my_module"
m_my_module.print_build_date();
```

Creating Apps Quickly

Creating an app from scratch can be a tedious task, especially for beginners. To address this there is a special add-on for the engine called app:

```
<!DOCTYPE html>
<html>
<head>
<script src="b4w.min.js"></script>
<script>

var m_app = b4w.require("app");
var m_data = b4w.require("data");

m_app.init({
    canvas_container_id: "container_id",
    callback: load_cb
})

function load_cb() {
    m_data.load("some_scene.json", loaded_cb);
}

function loaded_cb() {
    m_app.enable_camera_controls();
}

</script>
</head>

<body>
    <div id="container_id" style="width: 350px; height: 200px;"></div>
</body>

</html>
```

In this case the app module will create a `<canvas>` element inside the container with the specified `container_id` id. Then it will initialize the engine after the page is loaded and will finally execute the `load_cb()` callback.

Then the `some_scene.json` scene is loaded similar to the previous example. The only difference is that after the scene is loaded, the control system is initialized and camera movement with keyboard and mouse (or sensor screen) becomes possible.

In case when the app module is used, it is necessary to explicitly specify dimensions of the container element. Otherwise, the created `<canvas>` element will have zero dimensions.

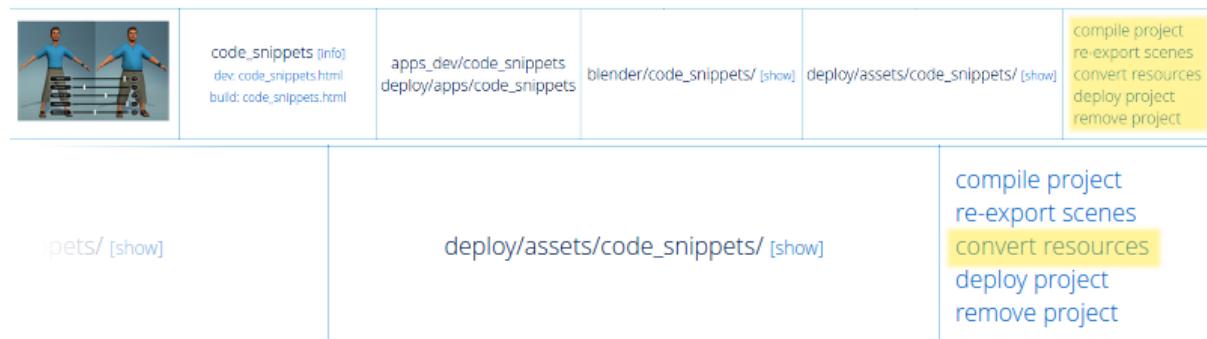
Resource Conversion

Currently, browsers do not fully support all possible media formats, so in order to create cross-browser applications (and also for optimization purposes) we recommend you to use the resource converter.

To support a wider range of platforms, a Python script (`scripts/converter.py`) for converting the source files into other formats is included into the distribution.

There are two ways to run this script.

Firstly, you can run it automatically using the project management system. The Convert Resources button can be found in the main page of the [Project Manager](#), in the Operations tab at the right side of the screen.



Secondly, you can run the script manually:

```
> cd <path_to_sdk>/scripts
> python3 converter.py [options] resize_textures | convert_dds | convert_media
```

For MS Windows users:

```
cd <path_to_sdk>\scripts
python converter.py [options] resize_textures | convert_dds | convert_media
```

Note: To run the scripts the Python 3.x needs to be installed in your system.

With the `-d` parameter you can specify the path to a directory in which converting will take place.

To exclude some directory from resource conversion, it is enough to put a file named `.b4w_no_conv` in it. This will not affect conversion in nested directories.

The `resize_textures` argument is used for decreasing texture resolution for the LOW mode.

Dependencies

Please make sure that you have all converter dependencies installed. You can do it with the following command:

```
> python3 <path_to_sdk>/scripts/converter.py check_dependencies
```

If some program is not installed, the following message will be displayed:

Couldn't find PROGRAM_NAME.

Linux

The list of dependencies is listed in the following table:

Name	Ubuntu 15.04 package
ImageMagick	imagemagick
NVIDIA Texture Tools	libnvtt-bin
Libav	libav-tools
FFmpeg	ffmpeg

Note: Linux users can additionally install the package qt-faststart which is used to optimize loading media files.

Windows

For MS Windows users it is not necessary to install these packages since they are already present in the SDK.

Mac OS

Mac OS users can install the brew package manager first and then install any missing packages.

Before installing packages, install the libpng and libjpeg libraries using these commands:

```
> brew install libpng  
> brew install libjpeg
```

Now you can proceed with installing required dependencies:

```
> brew install imagemagick  
> brew install --with-theora --with-libvpx --with-fdk-aac ffmpeg
```

In order to install NVIDIA Texture Tools, clone the repository with the following command:

```
> git clone https://github.com/TriumphLLC/NvidiaTextureTools.git
```

Now you can build and install the package:

```
> cd NvidiaTextureTools  
> ./configure
```

```
> make
> make install
```

Data Format

The conversion is performed as follows:

for audio (`convert_media`):

- ogg -> mp4
- mp3 -> ogg
- mp4 -> ogg

We recommend to use ogg as a base format. In this case the only conversion required for cross-browser compatibility will be ogg to mp4. Example of an input file: `file_name.ogg`, example of an output file: `file_name.altconv.mp4`.

for video (`convert_media`):

- webm -> m4v
- m4v -> webm
- ogv -> webm
- webm -> seq
- m4v -> seq
- ogv -> seq

We recommend to use WebM as a base format. In this case the only conversion required for cross-browser compatibility will be webm to m4v (webm to seq for iPhone). Example of an input file: `file_name.webm`, example of an output file: `file_name.altconv.m4v`.

for images (`convert_dds`):

- png -> dds
- jpg -> dds

Example of an input file: `file_name.jpg`, example of an output file: `file_name.altconv.jpg.dds`.

For the purpose of optimizing application performance it's possible to use min50 (halved) and DDS textures. In order to do this, we need to pass the following parameters during initialization of the application:

```
exports.init = function() {
    m_app.init({
        // ...
        assets_dds_available: true,
        assets_min50_available: true,
        // ...
    });
}
```

```

    });
    // ...
}

```

DDS textures require less memory (4 times less for RGBA data and 6 times for RGB data), but using them has following downsides:

- DDS textures might not work on some devices, especially the mobile ones, as not all of them support the WEBGL_compressed_texture_s3tc extension;
- as DDS is a lossy compression format, compression artifacts might be visible, especially on [normal](#) and [stencil](#) maps; it is recommended to [disable compression](#) for such textures.



An example of the DDS compression artifacts, particularly visible on the edges of the shadow.

During exporting the scene from Blender to the JSON format (but not the HTML format), DDS textures will be plugged in automatically, if they are present.

Textures can be converted to the DDS format using the [project manger](#) or the [scripts/converter.py](#) script described above. The .seq file format is used for sequential video. This is applied for IE 11 and iPhone because they are currently missing support for standard video formats for textures. Using dds format for images is more optimal compared to other formats.

The engine can use files which are manually created by a user if they have the following names: `file_name.altconv.m4v`, `file_name.altconv.mp3` and so on. Such files should be placed in the same directory as the media files used in Blender.

You can also use the free and cross-platform application [Miro Video Converter](#) to convert media files.

Code Examples

The SDK includes the Code Snippets application which demonstrates how to use the engine's functionality.

Currently, this application contains the following examples:

- Bone API - an example of individual bones position control
- Camera Animation - procedural camera animation
- Camera Move Styles - changing control modes for the camera
- Canvas Texture - working with canvas textures
- Custom Anchors - creating custom annotations
- Dynamic Geometry - procedural geometry modification
- Gyro (Mobile Only) - working with mobile devices' gyroscopes
- Instancing - copying scene objects in runtime
- Material API - tweaking material properties and replacing objects' materials
- Morphing - using shape keys
- Ray Test - the usage of raycasting for obstacles detection

The Code Snippets application is available at `SDK/apps_dev/code_snippets/code_snippets_dev.html`. It can be also run by using a link in the `index.html` file located in the SDK root.

Event-Driven Model

The event-driven model provides a universal interface for describing the 3D scene's change of state. It simplifies the processing of physics events and user actions.

Sensors

The basic unit of the event-driven model is a sensor. A sensor is a programming entity and can only be active (1, one) or inactive (0, zero). Some sensors may carry a payload which can be received in the manifold's callback function with corresponding API. For example, the ray-tracing sensor (Ray Sensor) provides the relative length of the intersection ray.

Users cannot directly control sensors via the external API. Instead, all sensors must be present in one or multiple collections - so called sensor manifolds. A manifold is a logic container associated with a scene object. It generates a response to a defined set of sensor events by executing a callback function. To define the manifold it is required to have the following information (see also the API documentation for description of the `controls.create_sensor_manifold` function):

- An object to carry the manifold (e.g. a thrown object).

- A unique id of the manifold (e.g. “IMPACT”).
- Callback execution mode (the options are: CT_POSITIVE - logic function positive result, CT_CONTINUOUS - every frame with a positive logic function result and once with a zero result, CT_LEVEL - any logic, function result change, CT_SHOT - one-stage logic function result change, CT_TRIGGER - logic function result switch, CT_CHANGE - any sensor value change).
- An array of sensors.
- A logic function to define the combination of the sensor states for which the callback function is executed.
- A callback function.
- An optional parameter to pass into the callback function.

You can read more about engine API in the [controls](#) module documentation.

Example

Let's consider the task to insonify the impact of a thrown stone. A distinctive sound should be produced for impacting different media (for example terrain and wall). There are collision meshes with physical materials in the Blender scene, material ids are “TERRAIN” and “WALL”. There is also a physical object being thrown in the scene, the object is named “Stone”.

Let's define a collision sensor for each medium, by the type of the sound produced.

```
// import the modules
var m_scenes = b4w.require("scenes");
var m_controls = b4w.require("controls");

// get the object being thrown
var stone = m_scenes.get_object_by_name("Stone");

// create the sensors
var sensor_impact_terrain = m_controls.create_collision_sensor(stone, "TERRAIN");
var sensor_impact_wall = m_controls.create_collision_sensor(stone, "WALL");
```

Add the sensors into an array. Use the OR logic in the logic function. Place the sound processing code in the callback function. Create the sensor manifold with the “IMPACT” id and the CT_SHOT type.

```
// array of the sensors
var impact_sens_array = [sensor_impact_terrain, sensor_impact_wall];

// manifold logic function
var impact_sens_logic = function(s) {return (s[0] || s[1])};

// callback
var impact_cb = function(obj, manifold_id, pulse) {
```

```
// NOTE: it's possible to play both sounds simultaneously

if (m_controls.get_sensor_value(obj, manifold_id, 0) == 1) {
    // ...
    console.log("play the terrain impact sound");
}

if (m_controls.get_sensor_value(obj, manifold_id, 1) == 1) {
    // ...
    console.log("play the wall impact sound");
}
}

// create the manifold
m_controls.create_sensor_manifold(stone, "IMPACT", m_ctl.CT_SHOT,
    impact_sens_array, impact_sens_logic, impact_cb);
```

When the “Stone” object collides with any physical material of “TERRAIN” or “WALL”, the callback function is executed. Inside this function we get the values of both sensors by their indices in the sensor array (0 - “TERRAIN”, 1 - “WALL”). The sensor value = 1 (active) means that the collision happened with the corresponding physical material. As a result the corresponding sound is produced (the code is not shown).

SDK File Structure

apps_dev source code of the applications

 code_snippets source files of the Blend4Web API usage demonstration application

 scripts Blend4Web API usage examples' source files

 dairy_plant source files of the Dairy Plant demo (available only in SDK Pro)

 fashion source files of the Fashion Show demo (available only in SDK Pro)

 firstperson source files of the Farm demo (available only in SDK Pro)

 flight source files of the Island demo

 new_year source files of the New Year 2015 greeting card

 project.py script for application developers

 victory_day_2015 source files of the V-Day 70 greeting card

 viewer the sources files of the Viewer application

 assets.json metadata with information about scenes loaded by the Viewer

 webplayer source files of the Web Player app

 website source files of applications from the Blend4Web official website

 blender source files of the Blender scenes

blender_scripts exporter and utility scripts for Blender
csrc source code (in C) of the binary part of the engine exporter and of the other utilities
deploy the resource directory for deploying on the server (scene source files, compiled
applications and documentation)
api_doc API documentation for developers (built automatically, based on the en-
gine's source code)
apps 3D applications intended for deploying; the directory duplicates apps_dev
common Compiled engine files. Shared by all applications from SDK (hence
the name).
assets downloadable resources: scenes, textures and sounds
doc the current user manual in HTML format, built automatically from doc_src
globals_detect utility code for detecting global variables
tutorials source files for the tutorials
doc_src source files of the current manual written in reST
index.html and index_assets main SDK webpage files
license files with license texts
Makefile makefile for building the engine, the applications, the documentation and for
deploying on a remote server (not available as a free version)
README.rst README file
scripts scripts
blend4web.lst, blend4web_sdk_free.lst and blend4web_sdk_pro.lst (optional)
lists of the files for building distributions
check_resources.py script for checking of and reporting about unused resources
(images and sounds referenced by the exported files)
compile_b4w.py script for building engine code and applications
converter.py script which halves the texture dimensions, compresses the textures
into the DDS format, converts sound files into mp4 and ogg formats
custom_json_encoder.py fork of the json Python module, sorts the keys in reverse
order
gen_glmatrix.sh script for generating the math module based on the source code
of glMatrix 2
graph.sh SVG generator for the current scene graph, used for debugging rendering
make_dist.py distributions builder script
memory.sh script for checking memory (RAM) and video memory (VRAM)
mod_list.py script for generating the list of modules to use in new applications

plot.sh debugging information graph builder
process_blend.py script for automatic reexport of all scenes from the SDK
remove_alpha_channel.sh script for removing the images' alpha channel
screencast.sh script for screen video recording
shader_analyzer.py script starting the local web server which calculates complexity of the shaders
translator.py script for building addon translations
shaders GLSL shaders of the engine
src main source code of the engine's kernel
addons source code of engine addons
ext source code of the external declarations that form the engine's API
libs source code of the libraries
tools Various tools for building the engine, apps or convert resources
converter_utils binary builds of the tools for resource conversion
closure-compiler Google Closure compiler, its externs and their generators
glsl
compiler compiler for the engine's GLSL shaders
pegjs grammars of the PEG.js parser generator for implementing the GLSL preprocessor, and also the script for generating the parser modules from these grammars
yuicompressor utility for compressing CSS files
uranium source code and building scripts of the Uranium physics engine (the fork of Bullet)
VERSION contains the current version of the engine

Loading Local Resources

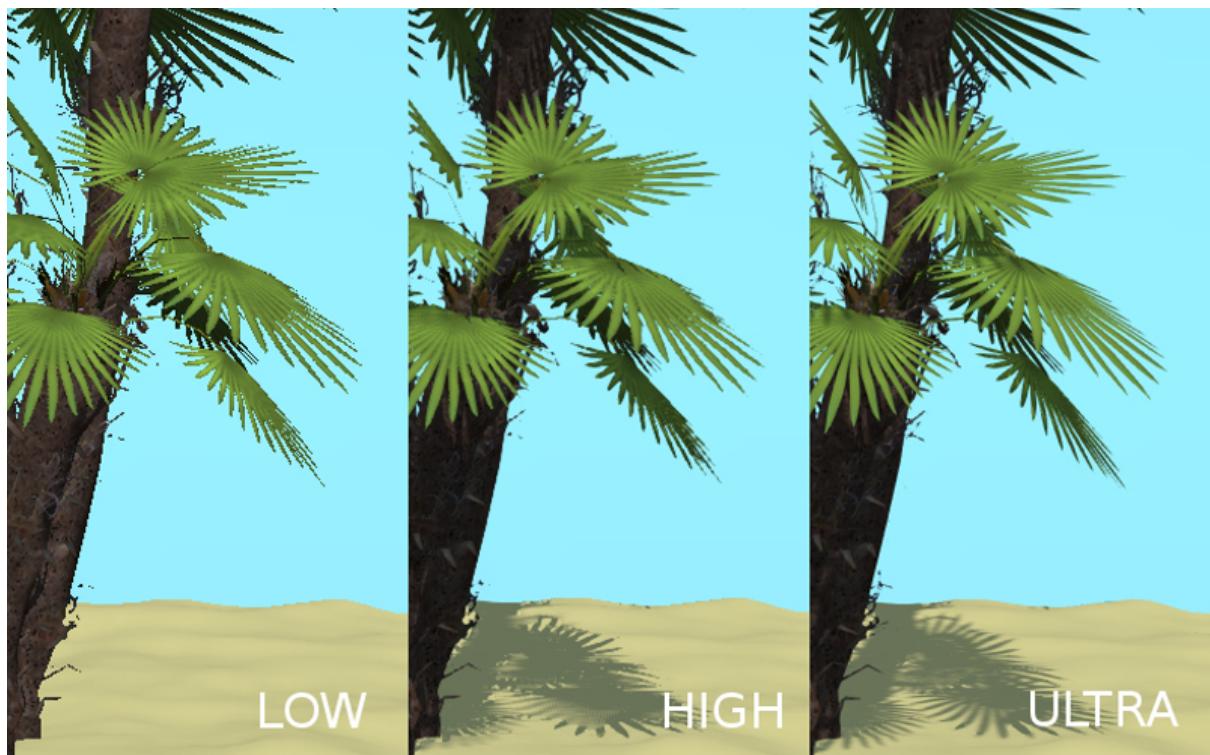
The engine's renderer is a web application and it works when you view an HTML file in a browser. After initialization the resources (scenes, textures) are loaded. This process is subject to the [same-origin policy](#) rule. In particular this rule forbids loading from a local directory.

Since version 15.02, the Blend4Web SDK includes the [development server](#) to solve the problem of loading local resources.

Quality Profiles

Several quality profiles are implemented in order to support platforms with different functionality.

- low quality (P_LOW) - a range of functions is turned off (such as shadows, dynamic reflection, postprocessing), the size of textures is halved when using a release version, anti-aliasing is disabled
- high quality (P_HIGH) - all features requested by the scene are used, the anti-aliasing method is FXAA
- maximum quality (P_ULTRA) - rendering resolution is doubled, resolution of shadow maps is increased, the anti-aliasing method is SMAA



Switching the quality profiles can be performed in runtime before initialization of the WebGL context. The default profile is P_HIGH.

```
var m_cfg = b4w.require("config");
var m_main = b4w.require("main");

m_cfg.set("quality", m_cfg.P_LOW);
m_main.init(...);
```

Application developers can also set the quality parameter upon engine initialization using

the app.js add-on:

```
var m_cfg = b4w.require("config");
var m_app = b4w.require("app");

m_app.init({
    canvas_container_id: "body_id",
    quality: m_cfg.P_HIGH
});
```

Non-FullScreen Web Apps

The Canvas element, to which the rendering is performed, can change its position relative to the browser window. This can occur due to some manipulations over the DOM tree, or as a result of page scrolling which is especially relevant for non-fullscreen web applications.

In most cases this will not affect the performance of the app by any means. However, some DOM events related to mouse cursor or touch position may carry incorrect information. This occurs because the coordinates obtained from the corresponding events are measured relative to the origin of the browser window, while the engine works with the coordinate space of the Canvas element itself (its origin is located in the top left corner).

1. If the top left corner of the Canvas element matches the top left corner of the browser window and is fixed in its position (non-movable) then it's sufficient to use event.clientX and event.clientY coordinates of the input events or get_coords_x() / get_coords_y() methods.

```
var m_mouse = require("mouse");

// ...
var x = event.clientX;
var y = event.clientY;
// ...
var x = m_mouse.get_coords_x(event);
var y = m_mouse.get_coords_y(event);
// ...
```

2. In the case of the scrolled browser window, you have to use event.pageX and event.pageY coordinates.

```
// ...
var x = event.pageX;
var y = event.pageY;
// ...
```

3. In the case of more sophisticated manipulations with the Canvas element (scrolling of the page elements, displacement from the top level corner of the browser window, changes in the DOM-tree) you need to perform correct coordinate conversions. In order to obtain coordinates suitable for use in the engine, you can convert them by using the client_to_canvas_coords() method of the container module:

```
var m_cont = require("container");
var _vec2_tmp = new Float32Array(2);
// ...
var canvas_xy = m_cont.client_to_canvas_coords(event.clientX, event.clientY, _vec2_tmp);
// ...
```

In order to obtain coordinates in the Canvas space, the engine should know its position relative to the browser window. However, if this position is subjected to changes during the work of the app (due to scrolling for example), the Canvas position should be recalculated. To do this automatically, you can set the `track_container_position` property upon app initialization:

```
exports.init = function() {
    m_app.init({
        // ...
        track_container_position: true,
        // ...
    });
    // ...
}
```

Please note, that this setting can lead to performance degradation in some browsers (such as Firefox) due to frequent DOM tree accesses. If the performance is critical, you can update the Canvas position manually when it is really necessary. To do this, use the `force_offsets_updating()` and `update_canvas_offsets()` methods instead of the `track_container_position` setting, or even the lower-level `set_canvas_offsets()` method from the container module:

```
var m_cont = require("container");
// ...
m_cont.force_offsets_updating();
// ...
m_cont.update_canvas_offsets();
// ...
m_cont.set_canvas_offsets(offset_left, offset_top);
// ...
```

4. The possibility to scale the whole webpage can lead to Canvas element displacement on mobile devices. The described decisions are suitable in this case. However, you can lock scaling at all and thus avoid such problems. It is enough to add the following meta-tag to the page's header:

```
<meta name="viewport" content="user-scalable=no">
```

For Engine Developers

Table of Contents

- For Engine Developers
 - Coding Style
 - * Examples
 - Building the Engine
 - Building the Addon
 - Dependencies
 - Naming Functions and Variables
 - Debugging
 - Shader Compilation
 - * Validation
 - * Obfuscation
 - * Optimization
 - * Import/Export Directives
 - * Recommendations and Limitations
 - * WebGL Extensions
 - * Compilation Errors
 - Updating Add-on Translations

Coding Style

This engine uses structural programming. The code is organized in modules. OOP methods are not used, classes are not defined, inheritance is not performed and so on.

The K&R style is used except for the fact that the opening bracket for a compound operator is placed on the same line, for example:

```
function foo_bar() {  
    // ...  
}  
  
if (a > b) {
```

```
// ...
}
```

4 spaces are used for indentation (no tabs allowed).

Examples

The underscore symbol is used in function and variable names:

```
var foo_bar = 123; // correct
var fooBar = 123; // wrong
```

All global variables begin with an underscore:

```
var _foo_bar = null;
```

The constants are written in capital letters and never begin with an underscore:

```
var FOO_BAR = 100;
```

The names of external API methods and properties are written after a point. To avoid obfuscation of fields they must be listed with the @cc_externs tag:

```
exports.FOO_BAR = 123;

exports.foo_bar = function() {

}

/**
 * Set properties.
 * @method module:properties.set_props
 * @param {Object} foo Foo object
 * @cc_externs props_1 props_2
 * @cc_externs props_3 props_4
 */
exports.set_props = function(foo) {

    var bar_1 = foo.props_1;
    var bar_2 = foo.props_2;
    var bar_3 = foo.props_3;
    var bar_4 = foo.props_4;

    ...
}
```

Commenting is in English only. Comment style - JSDoc.

Building the Engine

Before building please make sure that your system has all required dependencies installed (see the [table](#)).

To compile the engine and the applications included into the SDK please execute the following command (in the SDK root):

```
make compile
```

The full building that includes converting the resources (textures, sounds and videos), compilation and converting the docs, can be performed with the following command:

```
make build
```

Building the archives with the distributions:

```
make dist
```

All above mentioned operations can be performed with a single command:

```
make all
```

Building the Addon

Binary Blend4Web addon builds are available for the following platforms: Linux x32/64, OS X x64, Windows x32/64. At the same time users can compile the addon by themselves.

To do this Python 3.x (it's better if it's the same version as in Blender) and a C compiler are required. Under Linux it's enough to install the python3-dev and build-essential packages.

Paths relative to the repository root:

- build script: csrc/b4w_bin/build.py
- Blend4Web addon: blender_scripts/addons/blend4web/

The building process is started in the following way:

```
python3 ./csrc/b4w_bin/build.py
```

As a result of the building you'll get a binary file called:

b4w_bin_[PLATFORM]_[ARCHITECTURE].[STANDARD_EXTENSION],

located in the same directory as the addon. Example: b4w_bin_Linux_64.so. After this the addon is ready to use under this platform.

Dependencies

To do this Python 3.x (it's better if it's the same version as in Blender) and a C compiler are required. Under Linux it's enough to install the `python3-dev` and `build-essential` packages.

All dependencies are listed in the table below in order of decreasing importance.

Name	Ubuntu 14.04 package	Purpose
Bash	Included by default	interpreter for scripts
Python 3	Included by default	interpreter for scripts
NodeJS	nodejs	compiling shaders
Java	default-jre	compiling and obfuscating the engine modules
Emscripten	from EMSDK source code	building Uranium
ImageMagick	imagemagick	converting the resources
NVIDIA Texture Tools	libnvtt-bin	converting the resources
NVIDIA Cg Toolkit	nvidia-cg-toolkit	debugging shaders
Libav	libav-tools	converting the resources
Gnuplot	gnuplot	debugging
Graphviz	graphviz	debugging
xsel	xsel	debugging
Sphinx	sphinx-doc	building the manual (HTML version)
sphinx-intl	installed with PIP	building the manual (internationalization)
TeX Live	texlive, texlive-latex-extra texlive-lang-cyrillic	building the manual (PDF version)
JSDoc 3	from JSDoc source code	building the API documentation
PEG.js	from PEG.js source code	shader preprocessing

Naming Functions and Variables

When creating new functions and variables it is recommended to use the following prefixes and suffixes.

`init_` create an abstract object

`create_` create a certain object

`update_` update the state of an existing object

`attach_`/`detach_` add/remove a temporary object property

`append_`/`remove_` add/remove a temporary property to the already existing properties of the same kind

insert/_/pop_ add/remove an array element (accessed by index)
switch_ switch flag's binary value
apply/_/clear_ operation with flags, binary values or arbitrary parameters
set/_/get_ set/get the property/variable value
_tmp global variable - cache in the form of a simple object (array, vector)
_cache global variable - cache in the form of a complex object

Debugging

Engine debugging is performed with the debug.js module methods.

The structure of the current render graph can be saved in the DOT format using the b4w.debug.scenegraph_to_dot() call, for example, in the browser console. After calling this method, save the console's output into the file with the .gv extension. To get the graph in a visual form the [graphviz](#) utilities are required. Converting to the SVG format is performed using the command:

```
> dot -Tsvg graph.gv -o graph.svg
```

where graph.gv is the name of the file with the saved graph.

Shader Compilation

All shaders used in the engine are processed by a compiler. The compiler performs the following three main procedures:

- validation of the shader code,
- its obfuscation and
- optimization.

In order to run the compiler, execute one of the following commands in the SDK root:

```
> make compile_shaders  
> make verify_shaders
```

- make compile_shaders - performs validation, obfuscation, optimization and finally, export of the compiled shaders,
- make verify_shaders - performs only validation, obfuscation and optimization.

Syntax analysis (parsing) of the shader text is first performed during compilation. The corresponding parser is created automatically based on the grammar, using the [PEG.js](#) generator. Then the shaders are validated, obfuscated and optimized according to the parser data, and after that the shaders are exported in the form of an abstract syntax tree (AST) for direct loading in the engine.

The location of the main files in the repository:

- initial grammar - glsl_utils/pegjs/glsl_parser.pegjs
- parser generation script - glsl_utils/pegjs/gen_nodejs.sh
- parser - glsl_utils/compiler/glsl_parser.js

Validation

The compiler performs the following procedures related to shader code validation:

- reporting about unused variables and functions (dead code),
- checking the syntax of shaders,
- checking the conformance of shaders to the import/export mechanism,
- removing odd or repetitive tokens: spaces, endlines and semicolons.

Obfuscation

Obfuscation minifies the GLSL code and makes it difficult to understand it. So far the following procedure is implemented:

- replacing the user-defined identifiers with shorter single-symbol, two-symbol etc. names (with support of the import/export mechanism).

Optimization

Optimization constitutes the following procedures:

- removing curly brackets which are not useful in any ways except creating local scopes (this functionality is used for processing node/lamp directives),
- optimization inside functions - creating shared local variables to replace ones originally created by the programmer.

An example of removing unused curly brackets: replacing the following code

```
void function(){
    int a;
    {
        a = 1;
    }
}
```

with this code

```
void function(){
    int a;
```

```
a = 1;
}
```

Low number of temporary local variables is achieved by repetitively using them in different contexts. For example, the following code

```
int function(){
    int a = 1;
    int b = a + 3;
    return b;
}
```

will be replaced with

```
int function(){
    int _int_tmp0 = 1;
    _int_tmp0 = _int_tmp0 + 3;
    return _int_tmp0;
}
```

Note: Local variables for structures and arrays are not optimized this way.

Import/Export Directives

import/export directives are used to organize, structure and increase the readability of the shader code in the include file. They are specified in the beginning of the file and should look approximately like this:

```
#import u_frame_factor u_quatsb u_quatsa u_transb u_transa a_influence
#import qrot

#export skin
```

The `#import` directive defines a set of ids which are declared outside the include file but can be accessed from inside it. There is a limitation though: such ids must necessarily be declared somewhere above the place where the include file is linked.

The `#export` directive defines a set of ids which can be accessed from outside this file. Such ids must necessarily be declared in this file.

Therefore, the shader which uses the include file must have all the declarations necessary for import before the place of linking, and can use the exported ids after it.

Ids can be both variable names and function names. If there are no import/export directives it's considered by default that the include file does not use external declarations and does not allow the using of internal ones.

Recommendations and Limitations

Because of the following reasons: preprocessing, the need to process multiple shaders and include files and due to the compiler's features - it's possible to guarantee the work of the output code only if a number of rules and limitations are respected with regard to the shader source code:

1. In order to describe constants which are defined by the engine at run, it's necessary to use the `#var` special directive. For example:

```
#var AU_QUALIFIER uniform
AU_QUALIFIER float a;
```

The syntax here is similar to the `#define` directive. The point of the `#var` directive is that the value which it defines allows to parse the initial shader. It's irrelevant what exactly it will be (e.g. 'uniform' or 'attribute' in the above example), because at this level it's unknown anyway. Nevertheless, it's better to specify a more or less suitable description and not something arbitrary.

Note: The `#var` directive is not necessary for constants used not in the shader code but in the preprocessor expressions.

2. Using the import/export directives when needed.
3. The built-in functions must not be overloaded - only the user ones.
4. Variables should not be declared with names of the built-in functions, or main (even if it doesn't lead to errors).
5. The `#var` and `#define` directives must not be used for replacing single symbols in such operators as: “`++`”, “`-`”, “`*=`”, “`/=`”, “`+=`”, “`-=`”, “`==`”, “`<=`”, “`>=`”, “`!=`”, “`&&`”, “`||`”, “`^^`”.

For example:

```
#var EQUAL =
...
a *EQUAL b;
...
```

6. The usage of the `#include` directive should not lead to ambiguity during the obfuscation of the include file. This can happen when multiple shaders are included into the same file and the above defined directives (like `#var` or `#define`) can have influence on any of them. Also, it's better not to use undeclared functions and variables in the include file.
7. Multi-level includes or multiple inclusion of the same include into the same shader is not supported.
8. The shader's malfunction can also be caused by nontrivial using of preprocessing, for example, creating an invalid GLSL code:

```
#if TYPE
void function1() {
#else
void function1(int i) {
#endif
...
}
```

9. Do not declare variables with such names as node_[NODE_NAME]_var_[IN_OUT_NODE], where NODE_NAME — name of some node, IN_OUT_NODE — name of an input or an output of the node.
10. Repetitive use of #nodes_main, #nodes_global or #lamps_main directives is not permitted inside a single shader.
11. The #nodes_main, #nodes_global and #lamps_main directives are recommended to use in the file, containing these shader nodes description, for example, in the same include-file. This is necessary for the correct shader validation.

WebGL Extensions

Compilation may depend on WebGL extensions being used if they somehow influence the shading language. At the moment the following extensions are supported by the compiler:

- OES_standard_derivatives

Compilation Errors

In case of an error the compiler will output the corresponding message in the console.

Table of possible errors:

Error message	Cause
Error! Ambiguous obfuscation in include file ‘FILE_NAME’.	Ambiguous obfuscation in the ‘FILE_NAME’ include file.
Error! Bad preprocessing collision while obfuscation identifier: ‘NAME’. Varying/uniform or varying/attribute qualifiers combination. File: ‘FILE_NAME’.	Error in the FILE_NAME file. Its impossible to obfuscate the NAME variable because of redefining at preprocessing. Redefining the same variable with different qualifiers. Unacceptable combinations: varying/uniform, varying/attribute.
Error! Extension NAME is unsupported in obfuscator. File: ‘FILE_NAME’.	The NAME WebGL extension used in the FILE_NAME file is not supported by the obfuscator.
Error! Include ‘FILE_NAME’ not found.	The FILE_NAME include file could not be found.
Error! Undeclared TYPE: ‘NAME’. File: ‘FILE_NAME’.	Error in FILE_NAME file. Undeclared identifier NAME of type TYPE (variable, function, structure etc).
Error! Undeclared TYPE: ‘NAME’. Importing data missed. File: ‘FILE_NAME’.	Undeclared identifier NAME of type TYPE (variable, function, structure etc). Declaration missing for the identifier required in the FILE_NAME include file according to the #import directive.
Error! Undeclared TYPE: ‘NAME’. Possibly exporting needed in include file ‘INCLUDE_NAME’. File: ‘FILE_NAME’.	Error in FILE_NAME file. Undeclared identifier NAME of type TYPE (variable, function, structure etc). Possibly its export into the INCLUDE_NAME include file should be allowed.
Error! Undeclared TYPE: ‘NAME’. Possibly importing needed. File: ‘FILE_NAME’.	Undeclared identifier NAME of type TYPE (variable, function, structure etc). Possibly it should be specified as imported in the FILE_NAME include file.
Error! Unused export token ‘NAME’ in include file ‘FILE_NAME’.	Undeclared identifier NAME is allowed for export in the FILE_NAME include file.

Error! Using reserved word in TYPE 'NAME'. File: 'FILE_NAME'.	Error in FILE_NAME file. A reserved id is used for declaring the identifier NAME of type TYPE (variable, function, structure etc).
Error! 'all' extension cannot have BEHAVIOR_TYPE behavior. File: 'FILE_NAME'.	The #extension directive specified for all WebGL extensions in the FILE_NAME file does not support the behavior BEHAVIOR_TYPE.
Syntax Error. ERROR_MESSAGE. File: FILE_NAME, line: LINE_NUMBER, column: COL_NUMBER.	Syntax error in line LINE_NUMBER column COL_NUMBER during parsing the FILE_NAME shader. The initial error description is quoted in the ERROR_MESSAGE. The code listing taken from around the corresponding line is attached to the message (note the peculiarity of pegjs parser which specify the line which is a little bit after the actual error).
Warning! Function 'NAME' is declared in [include]file FILE_NAME, but never used.	An unused function NAME is declared in the FILE_NAME file.
Warning! Include file 'FILE_NAME' not used in any shader, would be omitted!	The FILE_NAME include file is not used in any of the shaders and so it will be excluded from the obfuscated version.
Warning! Unused import token 'NAME' in include file 'FILE_NAME'.	An unused id NAME is imported in the FILE_NAME include file.
Warning! Variable 'NAME' is declared in include file FILE_NAME, but never used.	An unused variable NAME is declared in the FILE_NAME file.

Updating Add-on Translations

If you need to update all existing .po files, run the script translator.py in the SDK/scripts directory without arguments:

```
> python3 translator.py
```

In order to update an existing .po file, run the script with a supported language code as an argument:

```
> python3 translator.py ru_RU
```

In order to view the list of supported languages, run the script as follows:

```
> python3 translator.py help
```

In any case, the file empty.po will be updated upon running the script.

After updates, the .po files can be edited/translated as usual.

Team Work. Using Git

Table of Contents

- Team Work. Using Git
 - Overview
 - Typical Workflow
 - Individual Settings
 - Checking the Status
 - Before the Commit
 - * Checking changes (of the text files)
 - * Rolling back files
 - * Unwanted files
 - Preparing for Commit
 - * Adding files
 - * Removing files
 - Commit
 - Syncing Between Repositories
 - * From the remote - to the local
 - * From the local - to the remote
 - Resolving Conflicts
 - * Overview
 - * The steps to be taken
 - * Binary files
 - * Text files
 - * Correcting commit
 - Tags
 - Other Useful Commands

Overview

In order to organize team work a Git version control system can be used. Using Git has a number of benefits as compared with other ways to collaborate:

- saving the history of changes with the possibility to roll back to previous versions
- synchronizing changes between users and automatic merging of changes
- working with large binary files is possible

Git is a distributed system and every developer or designer has his own local repository (storage). Syncing between the local repositories can be performed via the central (“shared”) storage, which can be located on a dedicated machine (server). Access to the server can be organized through SSH protocol.

Although there are many GUIs for Git beginners, here the work with the git standard console utility is explained.

Typical Workflow

1. Files can be created, added or deleted during the work process in the local repositories.
2. After a certain logical period of work is finished it is necessary to fix (commit) the changes and/or synchronize with your team mates.
3. Files are prepared for commit i.e. the accounting of changed, new and deleted files and also the resetting of changes.
4. Commit is performed.
5. Local changes are uploaded into the shared storage and become available for the colleagues.

A limited set of Git commands recommended for authoring applications and their graphical resources is described below.

It's necessary to switch to the repository before executing the commands, e.g.:

```
> cd ~/blend4web
```

Individual Settings

A new user can set up his name and email using the commands:

```
> git config --global user.name "Ivan Petrov"  
> git config --global user.email ipetrov@blend4web.com
```

The set up data will be used in the changelog.

Checking the Status

It's recommended to check the state of the repository before, in progress and after performing all the operations.

Use this command to check the status:

```
> git status
```

The result of the git status command if all the commits were performed and there are no new files:

```
# On branch master
# Your branch is ahead of 'origin/master' by 2 commits.
#
nothing to commit (working directory clean)
```

Possible result of git status if there are changes. For example the apps_dev/firstperson/firstperson.js and doc_src/git_short_manual.rst files are changed and a new file 123.txt is created:

```
# On branch master
# Changes not staged for commit:
#   (use "git add <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in working directory)
#
#       modified:   apps_dev/firstperson/firstperson.js
#       modified:   doc_src/git_short_manual.rst
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       123.txt
no changes added to commit (use "git add" and/or "git commit -a")
```

Before the Commit

Checking changes (of the text files)

In case of text files it is recommended to view the introduced changes before performing the commit.

Check what was changed in the whole directory:

```
> git diff
```

or in a specific file only:

```
> git diff apps_dev/firstperson/firstperson.js
```

A possible result of the git diff command for a text file:

```
diff --git a/apps_dev/firstperson/firstperson.js b/apps_dev/firstperson/firstperson.js
index 4381c99..44b3b15 100644
--- a/apps_dev/firstperson/firstperson.js
+++ b/apps_dev/firstperson/firstperson.js
@@ -557,8 +557,9 @@ function enable_camera_control_mode() {
    var cam_view_down = CAMERA_MOVE_UPDOWN * (Math.sin(_passed_time) - 1);

        b4w.camera.translate_view(obj, 0, cam_view_down, cam_view_angle);
- } else
+ } else {
    b4w.camera.translate_view(obj, 0, 0, 0);
+
}
```

Rolling back files

If the file was changed or deleted but it is necessary to recover it (to the latest committed state) use the command:

```
> git checkout doc_src/git_short_manual.rst
> git checkout 123.txt
```

The introduced changes will be canceled - this is why this command should be performed with caution.

Unwanted files

If a file is listed in the Untracked files (git status), but version control is not needed for it, it should be deleted or moved beyond the working directory.

Preparing for Commit

Adding files

If you are happy with the changes, add the needed changed and/or new files for commit.

```
> git add apps_dev/firstperson/firstperson.js
> git add 123.txt
```

Check the status again:

```
> git status
```

Possible result of the git status command after adding some files with the git add command:

```
# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#   new file:  123.txt
#   modified:  apps_dev/firstperson/firstperson.js
#
# Changes not staged for commit:
#   (use "git add <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in working directory)
#
#   modified:  doc_src/git_short_manual.rst
#
```

You can see that the apps_dev/firstperson/firstperson.js and 123.txt files were added for commit and the doc_src/git_short_manual.rst file was not added. To make things easier it is recommended to either add such files for commit or cancel their changes with the git checkout command.

Removing files

Some files can be marked as deleted from Git after performing the git status command, for example:

```
# On branch master
# Your branch is ahead of 'origin/master' by 2 commits.
#
# Changes not staged for commit:
#   (use "git add/rm <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in working directory)
#
#   deleted:   123.txt
#
no changes added to commit (use "git add" and/or "git commit -a")
```

In this case if deleting the file should be recorded (i.e. enter the commit), perform the git rm command, for example:

```
> git rm 123.txt
```

If the file was deleted by accident and its necessary to recover it, use the git checkout command.

Commit

Perform commit with the command:

```
> git commit
```

A text editor window will show up (for example, nano or vim), in which it's necessary to enter the commit comment in English.

GNU nano 2.2.6	File: .git/COMMIT_EDITMSG		
My commit message			
# Please enter the commit message for your changes. Lines starting			
# with '#' will be ignored, and an empty message aborts the commit.			
# On branch master			
# Changes to be committed:			
# (use "git reset HEAD <file>..." to unstage)			
#			
# new file: 123.txt			
# modified: apps_dev/firstperson/firstperson.js			
#			
# Changes not staged for commit:			
# (use "git add <file>..." to update what will be committed)			
# (use "git checkout -- <file>..." to discard changes in working directory)			
#			
# modified: doc_src/git_short_manual.rst			
#			
^G Get Help	^O WriteOut	^R Read File	^Y Prev Page
^X Exit	^J Justify	^W Where Is	^V Next Page

Save the changes and quit the editor (in nano Ctrl+O, then Ctrl+X; in vim ZZ, or ESC :wq).

After commit it's recommended to recheck the status. Commit is performed correctly if the git status command returns nothing to commit, working directory clean.

Syncing Between Repositories

From the remote - to the local

After all the commits are performed it's necessary to load the changes from the remote ("shared") repository to the local one:

```
> git pull
```

Result of the git pull command if there are no changes in the remote repository:

Already up-to-date.

Result of the git pull command if the remote repository contains changes and syncing was successful:

```
remote: Counting objects: 151, done.
remote: Compressing objects: 100% (101/101), done.
remote: Total 102 (delta 74), reused 0 (delta 0)
Receiving objects: 100% (102/102), 69.77 MiB | 4.87 MiB/s, done.
Resolving deltas: 100% (74/74), completed with 32 local objects.
From lixer:blend4web
  dbf3877..9f9700c master -> origin/master
Updating dbf3877..9f9700c
Fast-forward
  apps_dev/firstperson/firstperson.js      | 338 ++
  .../location_agriculture.blend          | Bin 25601626 -> 25598644 bytes
...
  src/controls.js                         |  38 ++
  src/data.js                            |   5 +
  src/physics.js                         | 185 ++
19 files changed, 1452 insertions(+), 2767 deletions(-)
create mode 100644 deploy/assets/location_agriculture/textures/rotonda_02_diff.png
```

If you wish it's possible to look up the changes made by your colleagues using the following command:

```
> git diff dbf3877..9f9700c
```

The parameter of this command - in this case dbf3877..9f9700c - shows between which commits exactly the changes were made. This parameter can be conveniently selected in the console in the git pull results and pasted with a mouse click (middle button) where you need.

You can also view the changelog:

```
> git log
```

The git pull command does not always lead to a successful synchronization. The result of git pull when there are conflicts:

```
remote: Counting objects: 11, done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 6 (delta 5), reused 0 (delta 0)
Unpacking objects: 100% (6/6), done.
From lixer:blend4web
  ff715c2..dbf316a master -> origin/master
warning: Cannot merge binary files: blender/landscape_objects/Fallen_tree.blend (...)

Auto-merging blender/landscape_objects/Fallen_tree.blend
CONFLICT (content): Merge conflict in blender/landscape_objects/Fallen_tree.blend
Automatic merge failed; fix conflicts and then commit the result.
```

The steps to be taken at conflicts are described below.

From the local - to the remote

After that the changes should be uploaded from the local repository to the remote ("shared") one to make the changes available for team mates.

```
> git push
```

The result of the git push command if the remote repository already contains all the local changes:

```
Everything up-to-date
```

The result of the git push command if synchronization was successful:

```
Counting objects: 25, done.  
Delta compression using up to 8 threads.  
Compressing objects: 100% (14/14), done.  
Writing objects: 100% (14/14), 1.23 KiB, done.  
Total 14 (delta 11), reused 0 (delta 0)  
To gfxteam@lixer:blend4web.git  
 9f9700c..fa1d6ac master -> master
```

The result of the git push command if synchronization was not successful because the git pull command was not executed first:

```
To gfxteam@lixer:blend4web.git  
! [rejected]      master -> master (non-fast-forward)  
error: failed to push some refs to 'gfxteam@lixer:blend4web.git'  
To prevent you from losing history, non-fast-forward updates were rejected  
Merge the remote changes (e.g. 'git pull') before pushing again. See the  
'Note about fast-forwards' section of 'git push --help' for details.
```

You should execute the git pull command.

The changes uploaded into the central repository can be received by other developers with the git pull command.

Resolving Conflicts

Overview

Synchronization conflicts occur if both conditions are met

1. the same file was changed both in the local and remote repositories, and
2. automatic merging of the changes didn't occur because the changes are in the same place of the file.

Typical cases:

1. a binary file (texture, blend file) was independently changed by two developers
2. different changes were introduced to the same line of the same text file
3. one developer has changed the file while the other has moved it and so on.

Although synchronization conflicts are normal, if they happen too often it slows down the work. It is recommended to notify your team mates about start of working with the shared binary files, and also to perform synchronization more often. It is necessary to effectively distribute the work between developers to reduce the number of such shared files. This can be achieved particularly through linking of all the scene's resources from the separate blend files into the master file.

The steps to be taken

It's not recommended to perform any files operations (modifying, deleting) while the repository is in a conflict state.

The first thing to do is to perform the git status command.

```
# On branch master
# Your branch and 'origin/master' have diverged,
# and have 7 and 1 different commit each, respectively.
#
# Unmerged paths:
#   (use "git add/rm <file>..." as appropriate to mark resolution)
#
#   both modified:   blender/landscape_objects/Fallen_tree.blend
#
no changes added to commit (use "git add" and/or "git commit -a")
```

A list of conflicting files can be found in the Unmerged paths section.

The order of the following steps is different for binary and text files.

Binary files

At this stage the conflicting binary files are in the same state as they were in the local repository before the synchronization attempt. The files are fully functional (for example they can be opened by graphics editors).

In case of conflicting binary files it's necessary to sort out (with the team mates or by yourself) which of the files should be left and which should be discarded. Selecting can be performed with the git checkout command.

Select the local version of the file (- -ours). To make sure that it's local you can open it.

```
> git checkout --ours blender/landscape_objects/Fallen_tree.blend
```

Select the remote version of the file (- -theirs). To make sure that it's remote you can open it.

```
> git checkout --theirs blender/landscape_objects/Fallen_tree.blend
```

Select the local version of the file again (- -ours).

```
> git checkout --ours blender/landscape_objects/Fallen_tree.blend
```

Eventually you have to stick to the right version of the file. In case there is a threat of losing the work you may save the discarded file outside the repository.

Text files

At this stage Git introduces both local and remote changes to the conflicting text files, in a special format. Such text files are not workable as a rule

Example. One developer changed the scene name from “Blue Lizard” to “Green Lizard” in the application file and uploaded the changes into the central repository. Another developer changed “Blue Lizard” to “Red Lizard” in the same line, performed commit and executed the git pull command. As a result this very developer will be responsible for resolving the conflict. The following lines will be present in his version of the application file:

```
<<<<<< HEAD
    "name": "Red Lizard",
=====
    "name": "Green Lizard",
>>>>> 81bf4e2d5610d500ad4d2a2605ee7e61f759f201
```

In case of conflicting text files the following steps can be taken. Files with source code should be edited with or without respect to the changes introduced by both parties. On the other hand, it is easier to reexport the exported scene text files (ending with .json).

Correcting commit

After selecting the required files or editing the changes, add them for commit:

```
> git add blender/landscape_objects/Fallen_tree.blend
> git status
```

Possible result of git status command after adding the conflicting files for commit:

```
# On branch master
# Your branch and 'origin/master' have diverged,
# and have 7 and 1 different commit each, respectively.
#
nothing to commit (working directory clean)
```

Perform commit. It is recommended to leave the default comment:

```
> git commit  
> git status
```

```
# On branch master  
# Your branch is ahead of 'origin/master' by 8 commits.  
#  
nothing to commit (working directory clean)
```

Conflicts are resolved, the changes from the remote repository are successfully applied in the local repository. Now the changes in the local repository - including the just resolved conflict - can be uploaded to the remote repository with the git push command.

Tags

Tags are intended for pointing at a certain commit, for example, to specify a stable product version.

View the list of tags:

```
> git tag
```

Create a tag for the release from June 3, 2013, pointing to the commit with a stable product version:

```
> git tag R130603 67bb597f7ed1643ed0220d57e894f28662e614e5
```

Check the commit tag information:

```
> git show --shortstat R130603
```

Roll back to the tag...

```
> git checkout R130603
```

...and return:

```
> git checkout master
```

Synchronize the tags with the remote repository:

```
> git push --tags
```

Delete the tag (if created by mistake):

```
> git tag -d R130603
```

Other Useful Commands

Check the log for January, 2012, show file names without merging commits:

```
> git log --after={2012-01-01} --before={2012-01-31} --name-only --no-merges
```

Problems and Solutions

Table of Contents

- Problems and Solutions
 - WebGL Support
 - Problems Upon Startup
 - WebGL Failed to Initialize
 - More In-Depth Troubleshooting
 - Known Issues

As WebGL is still a relatively new technology, it may not work perfectly with every combination of software and hardware. This chapter covers common problems that users of the Blend4Web engine may encounter and provides solutions for these problems.

WebGL Support

If you are using a desktop or laptop computer, your system must have a GPU that supports DirectX 9.0c and OpenGL 2.1, such as:

- Nvidia GeForce 6xxx series or higher.
- AMD/ATi Radeon R500 (X1xxx) series or higher.
- Intel GMA 950 or higher.

If you are using WebGL on a mobile device, please check whether your device is on the compatibility list.

You also need to have a web browser that supports WebGL technology.

The following web browsers support WebGL:

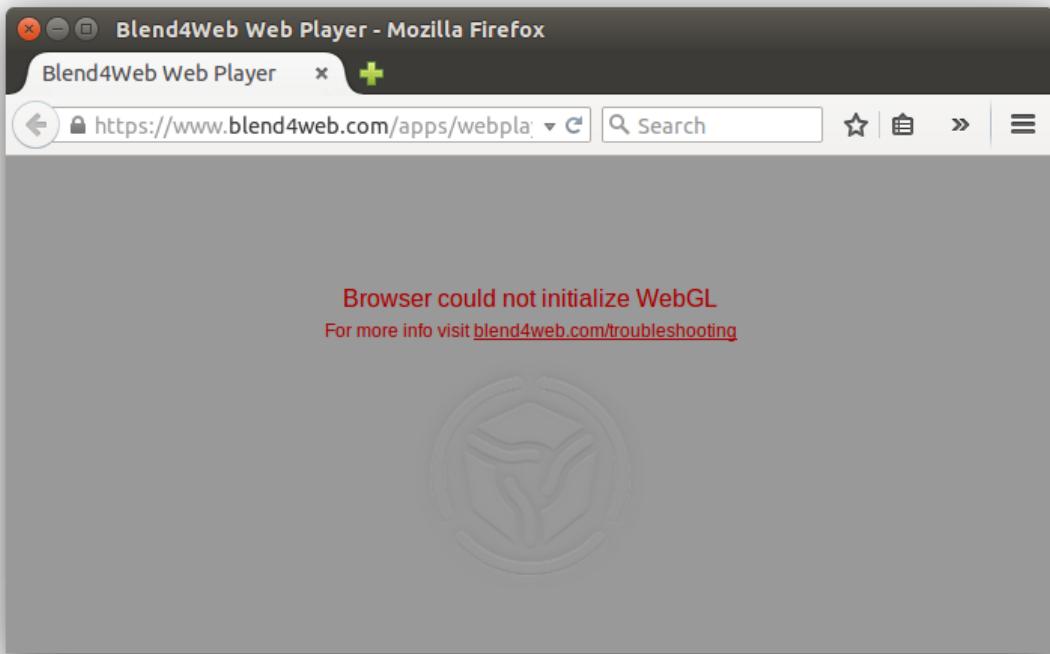
- Google Chrome (v. 9 or higher)
- Mozilla Firefox (v 4.0 or higher)
- Safari (v. 8.0 or higher)

- Chrome for Android (v. 25 or higher)
- Internet Explorer (v. 11 or higher)
- Microsoft Edge
- Opera (v. 12 or higher)
- UC Browser (experimental)

We also recommend to use the most recent version of the web browser to avoid compatibility problems.

Problems Upon Startup

1. The “Browser could not initialize WebGL” message is shown.



Follow the instructions listed in the [WebGL Failed to Initialize](#) section.

2. The user interface or background is shown but the default scene is not rendered. At the same time the <http://get.webgl.org/> site and other WebGL applications are working correctly.

Possible causes:

- The engine tries to load resource files which were moved or deleted.
- You are using the old versions of video drivers.
- You are using open source drivers which do not fully support WebGL.

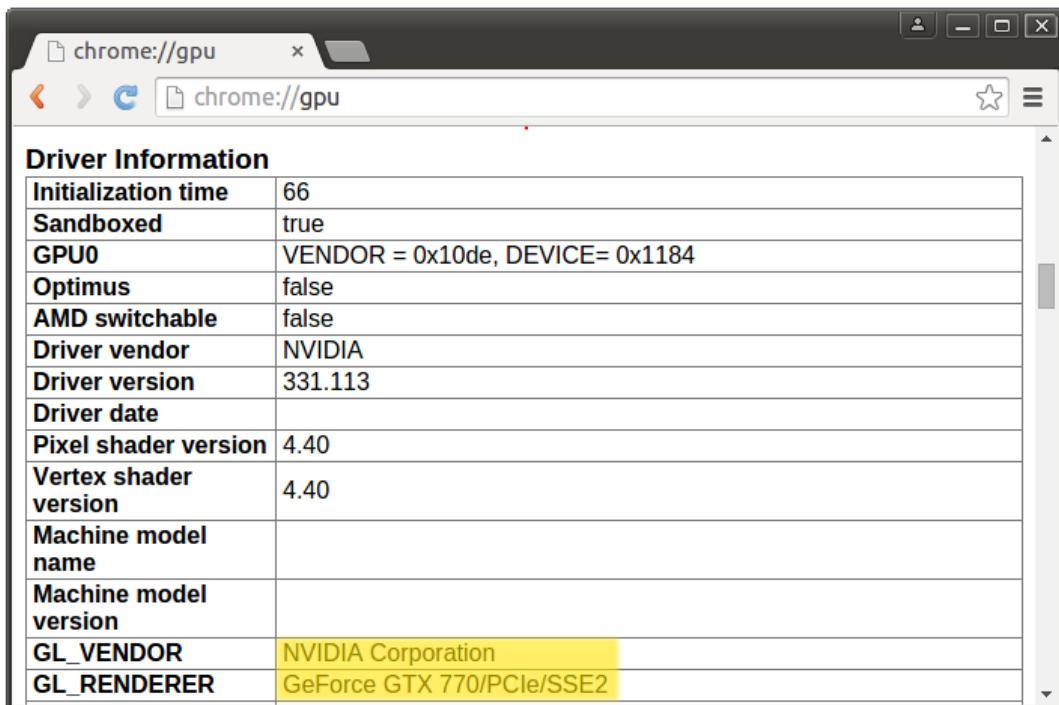
For Linux users - due to incomplete OpenGL implementation in open source drivers at the moment it is recommended to use current versions of proprietary drivers for Nvidia and AMD video cards.

- You are using an outdated operating system, such as Windows XP.
- Browser is not set up for loading local resources. In this case, the problem can be fixed by using local web server. See the [Loading Local Resources](#) section.

WebGL Failed to Initialize

The <http://get.webgl.org/> page tells about problems when viewing it in recent Chrome or Firefox. What can I do?

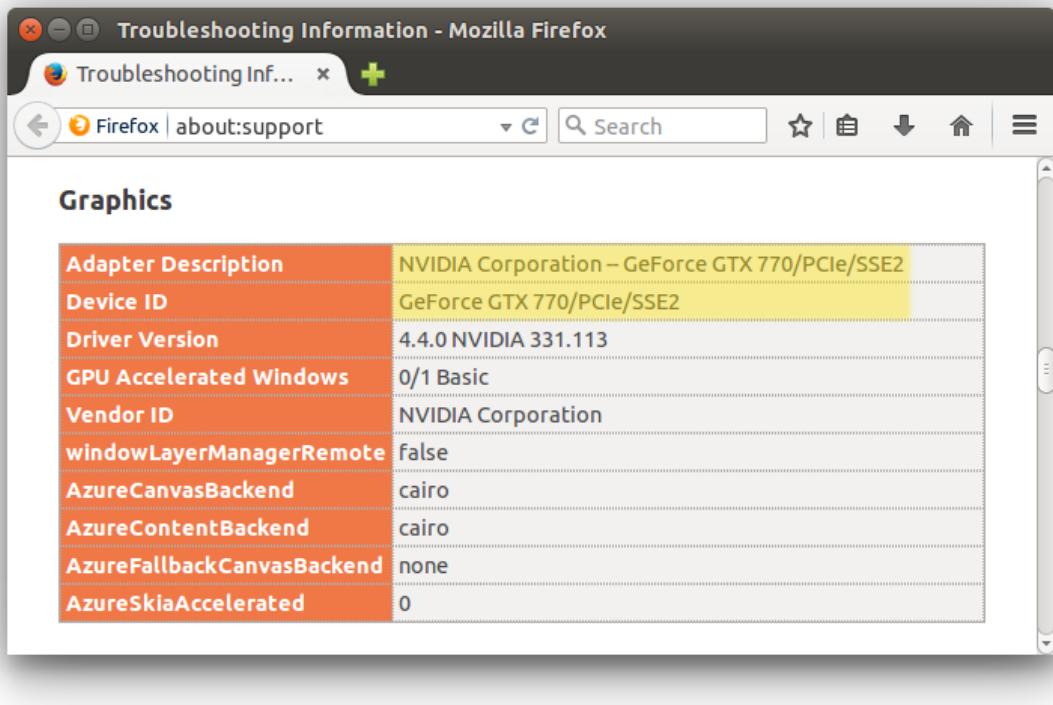
1. Install the latest updates for your system (for MS Windows see [the guide](#)). In case of MS Windows install the latest [DirectX runtime](#). Reboot.
2. It is recommended to timely update video card drivers. To detect your video card and its vendor please type `about:gpu` (or `chrome://gpu`) to the address bar of Chrome browser...



A screenshot of a Chrome browser window. The address bar shows "chrome://gpu". The main content area displays a table titled "Driver Information" with the following data:

Initialization time	66
Sandboxed	true
GPU0	VENDOR = 0x10de, DEVICE= 0x1184
Optimus	false
AMD switchable	false
Driver vendor	NVIDIA
Driver version	331.113
Driver date	
Pixel shader version	4.40
Vertex shader version	4.40
Machine model name	
Machine model version	
GL_VENDOR	NVIDIA Corporation
GL_RENDERER	GeForce GTX 770/PCIe/SSE2

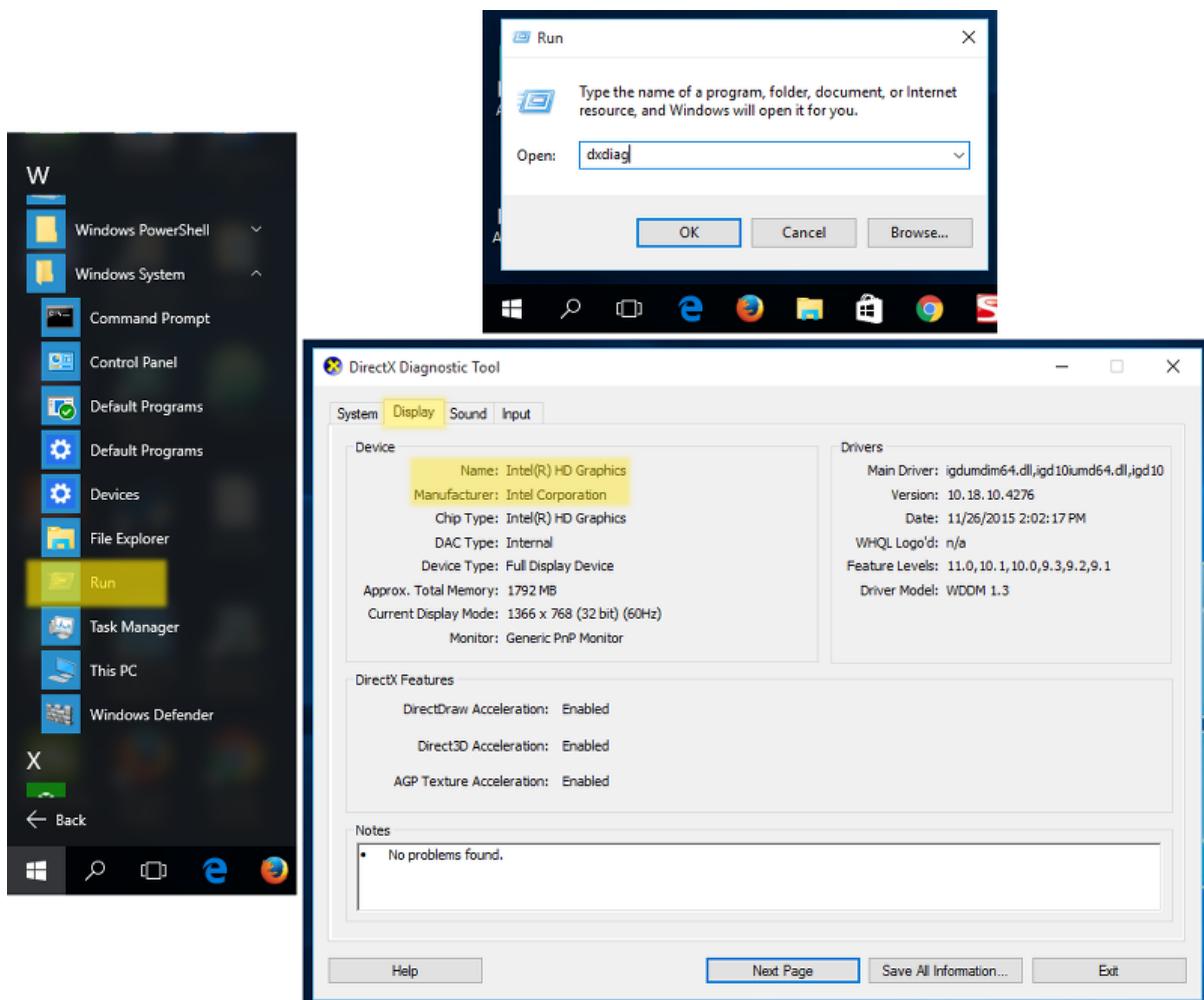
or Firefox...



For Windows, you can run the DirectX Diagnostic Tool called dxdiag.

To do it, please follow these steps:

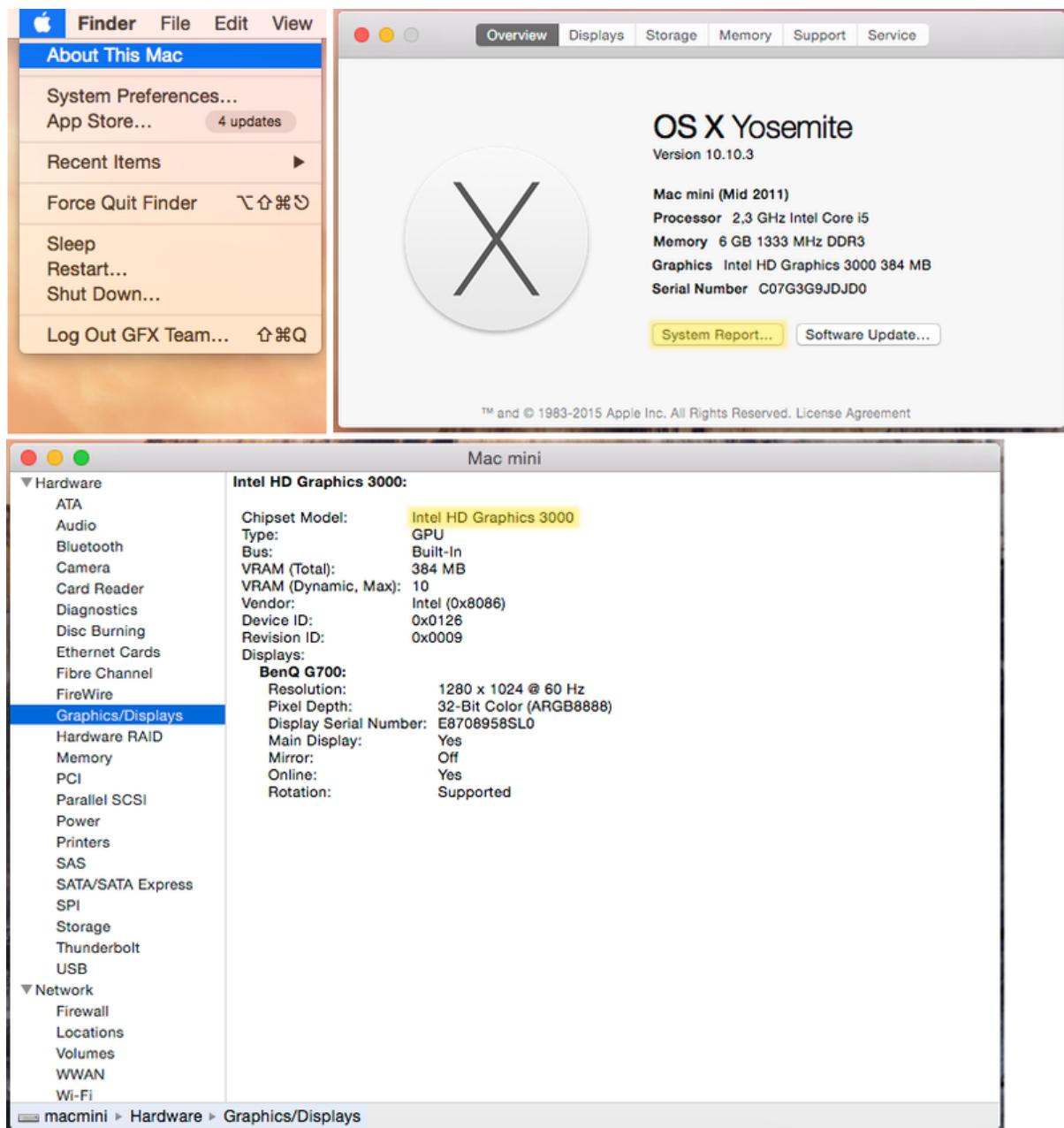
1. Select the Run command from the Start menu
2. Type dxdiag to the Open field and press Enter to open DirectX Diagnostic Tool
3. Open the Display panel. There you can find manufacturer, model and other information regarding your video card.



For MacOS X, you can check System Report.

To do it, please follow these steps:

1. Select About This Mac from the Apple menu.
2. Click System Report button.
3. Select Graphics/Displays in the Hardware section.

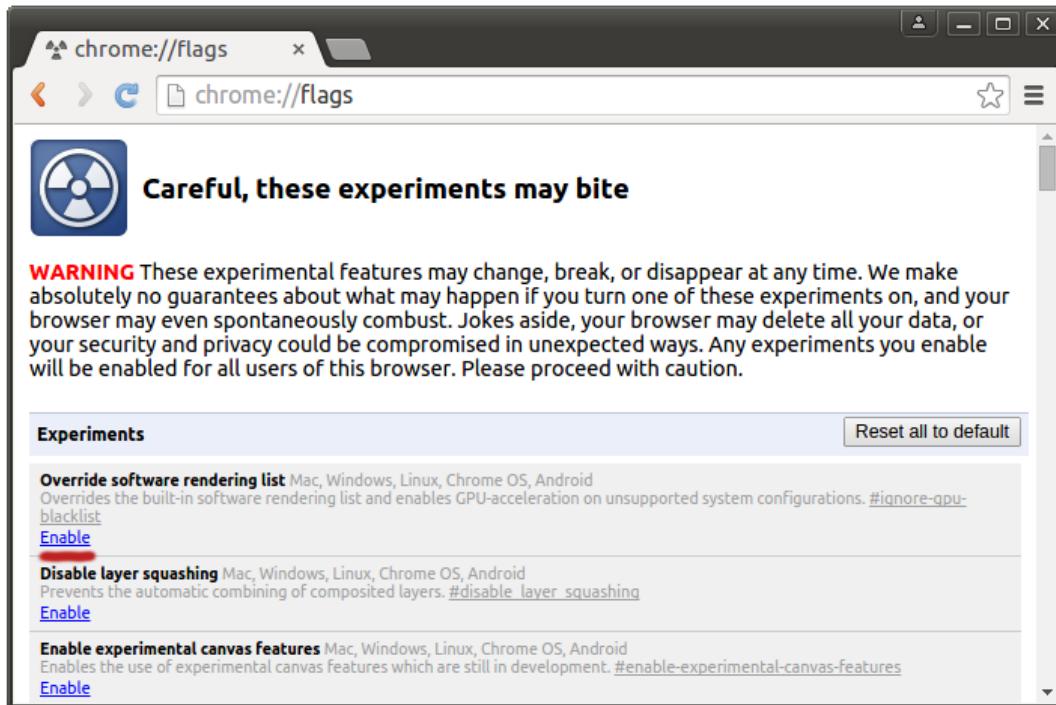


Download the drivers from the corresponding support center (for example Intel, Nvidia, AMD/ATI). Reboot the system after the drivers are installed.

3. If the measures described above did not help to initialize rendering (or there is no possibility to update the system) try to change the browser settings.

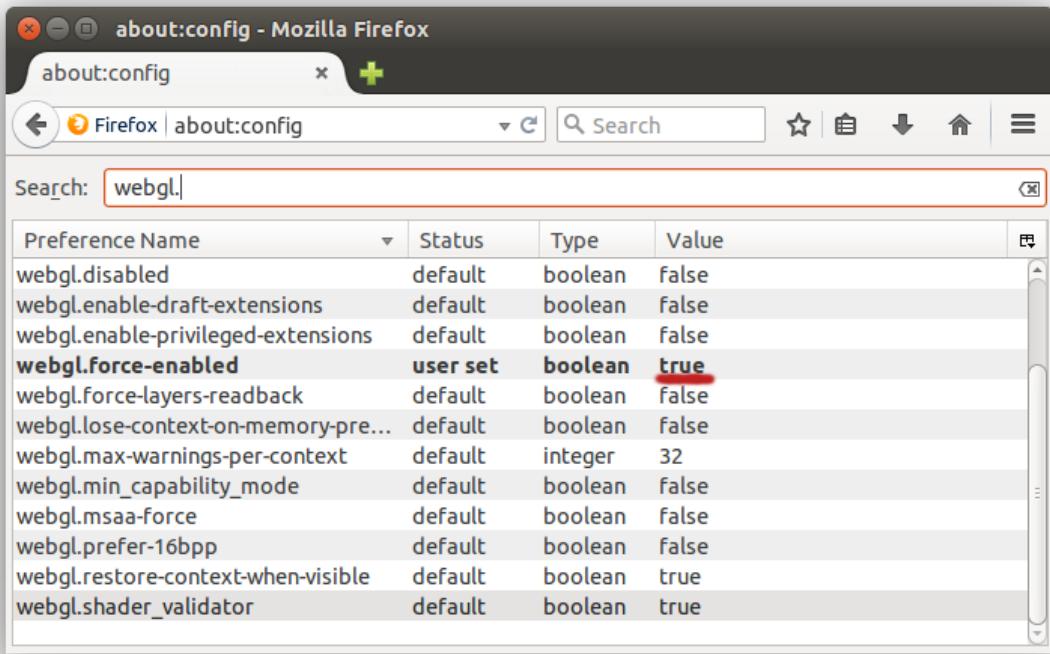
For Chrome:

Enter `about:flags` (or `chrome://flags`) into the browser's address bar, click **Enable** under the **Override software rendering list** option and restart the browser.



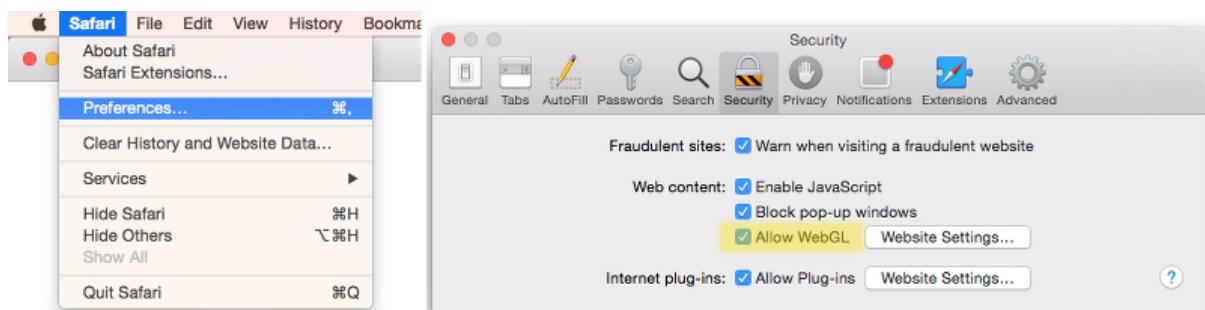
For Firefox:

Enter about:config into the browser's address bar, search for the webgl.force-enabled parameter and double-click on it to switch from false to true.



For Safari

Select Preferences from Safari menu, select the Security tab and make sure that Allow WebGL checkbox is enabled.



More In-Depth Troubleshooting

If nothing mentioned above helped you solve the issues you are experiencing, please visit the Blend4Web [community forum](#) and leave a message in the [Bug Report](#) thread. Our team will be sure to help you.

Known Issues

- Problems with updating of the add-on.

It's strongly advised to restart Blender after installing a newer version of Addon/SDK.

- NVIDIA 331 driver in Linux can cause WebGL errors.

- Changed texture filtering on some platforms.

An incorrect texture filtering was disabled on iPad and Internet Explorer for materials with Alpha Clip type of transparency.

- Some devices with Mail GPU require manual WebGL activation in browser settings.
- For the local development server to work on Apple OS X and Blender 2.76, you may need to install Python 3.4 <<https://www.python.org/downloads/release/python-343/>>. This is due to a bug in Blender <https://developer.blender.org/T46623>. This bug has been fixed in Blender 2.76b, so updating it is advised.
- Skeletal animation may work incorrectly while using Nouveau drivers.
- Transparent textures may not render correctly in the IE11 and Microsoft Edge web browsers and on iPad.

The problem is relevant for the [transparent materials](#). Image artifacts are visible in the areas where alpha channel value is close or equal to zero. To fix this issue, it is recommended to increase the value of the alpha channel until artifact are no longer visible (increasing it by value from 0.01 to 0.05 should be enough in the most cases).

- WebGL crashes on Linux Chromium with Nvidia GeForce 400/500 series GPUs with drivers later than 355.

This issue is caused by incompatibility of Chromium sandbox and NVIDIA's latest drivers. The solution is to downgrade drivers to the 340xx version.

- WebAudio issues.

Audio doesn't work in some versions of Google Chrome for the "Background Music" speakers in case of HTML export. Currently, speaker type is automatically changed to "Background Sound" as a workaround.

There is an audio context error when running too many instances (>6) of b4w engine (for example, many browser tabs or many iframes on one page) in Google Chrome. The error is prevented by disabling the audio for an application if the audio context cannot be created for it.

- QQ Browser doesn't support WebGL at the moment.

- WebGL is unstable and sometimes crashes on Mesa 11.x drivers in Linux/Chrome with Intel GPUs. Downgrading to Mesa 10.x drivers can help.

Release Notes

v16.05

New Features

- Experimental HMD configurator.

Add-on `hmd_conf.js` has been added. It allows users to set up HMD parameters manually. For now, this configurator has been implemented in the Viewer app and can be executed by the button located on the Stereo View panel. For more information see API documentation on the `hmd_conf` module.

- Experimental configurator for gamepads and other game controllers.

Add-on `gp_conf` has been added to API. It contains two main functions: `show` and `hide`. For now it supports controllers and steering devices.

To test this configurator in the Viewer app activate the Gamepad Settings check box on the Input Devices panel.

- New code snippets.

New code snippet demos have been added. Their names are Gamepad, Webcam and Change Image. The first one can help you to integrate gamepads into your project, the second shows you the Camera API in action and the third one is the implementation of the new texture functions.

- A new Vector Transform material node.

This node can be used to convert input data between object, world and camera coordinate spaces.

- A new Empty logic node.

This node can be used to simplify rerouting complex logic node configurations.

- Dynamic replacement of texture images.

A new method `change_image` has been added. It's possible now to replace textures and cubemap bitmaps via API.

- Batching based on a new clustering algorithm.

Batching, that is, the process of combining similar objects for performance reasons, is now based on a new clustering algorithm applied at export. This improves the whole batching process by making it "smarter" and more optimized in comparison with the old regular grid batching.

- Project Manager improvements.

To maintain naming consistency all command line options in `project.py` utility are now specified with `-` symbol instead of underscore, e.g `--engine_type` option has become `--engine-type`.

Added the new `--assets-dest` command property to `project.py` utility. This option allows users to specify a destination directory for storing assets in deployed projects.

- New materials in Material Library

4 new materials have been added to the Pro SDK Material Library: Gold, Ribbed Glass, Silk and Velvet.

Changes

- UI and export usability improvements for objects with no option from the Export Options list selected.

`Export Shape Keys` is selected automatically after adding a shape key.

`Export Vertex Animation` is selected automatically after baking vertex animation.

`Apply Scale and Modifiers` is automatically used during the export process for non-uniform scaled objects, which meet the following requirements:

Has no vertex animation.

Has no parent object.

Has no skinning.

Object physics is disabled.

`Apply Modifiers` is automatically used during the export process for objects, which have modifiers and meet the following requirements:

Has no vertex animation.

Has no skinning.

- Z sorting in Alpha Sort materials is now performed based on the sizes of objects.

This can lead to more frequent sorting updates which impairs performance, but fixes sorting issues on small objects.

- New frustum culling algorithm.

Now we calculate frustum culling using specific mesh materials instead of objects.

- Gamepad sensor changes.

New button and axis identifiers have been added to the `input` module. Also, gamepad sensors have been renamed as `create_gamepad_btn_sensor` and `create_gamepad_axis_sensor`.

- Entry Point logic nodes with the Run From Script option selected can now be called from API multiple times.

- API changes.

The `create_rotation_preloader` method of the `preloader` module has been declared deprecated and will be removed in further releases.

The `create_advanced_preloader` method of the `preloader` module has been declared deprecated and will be removed in further releases.

The `set_config` method has been added to the `input` module.

The `register_device` method of the `input` module has been declared deprecated and will be removed in further releases.

The `gyro_use` flag of the `config` module has been removed (not needed anymore).

Now functions of the `storage` module have an optional last parameter.

The `is_ie11` method has been added to the `util` module.

- Now the state of a keyboard button is stored in the payload of the keyboard sensor.

For more information see API documentation on the `create_keyboard_sensor` method.

- The link to the troubleshooting page in the WebPlayer “Browser could not initialize WebGL” message has been changed to a more appropriate one.

Now it refers one to the “Problems and Solutions” page in the documentation.

- The documentation has been considerably expanded with detailed descriptions for Anchor and Viewport Alignment features.

Fixes

- Fixed issue with incorrect automatic updates in addon.
- Fixed anchors behavior with logic nodes.
- Fixed JS Callback logic node obfuscation.

- Fixed issue with incorrect sensor removal in `controls.remove_sensor_manifolds` method.
- Fixed issue with incorrect age and compression ratio in distribution zip files.
- Fixed multi-sampling issue in Firefox browsers.

Disabled partially supported multi-sampling in Firefox WebGL 2.0 implementation.
- Fixed error when canvas resolution wouldn't change when Anti-Aliasing quality settings were changed.
- Fixed error when the diffuse intensity value would be applied to shadeless materials.
- Fixed engine crash in the case when `B4W_GLOW_OUTPUT` and `B4W_REFRACTION` nodes were used in the same material.
- Fixed the duplication of the Custom Properties panel in Blender's interface for the Font, Curve, Lattice, Armature and MetaBall data types.
- Fixed some errors in the compiled version of the engine.
- Fixed issues with input and controls modules.
- Fixed issue when node materials using the ColorRamp node were incorrectly batched.
- Fixed incorrect HTTP server shutdown.

v16.04

New Features

- Support for gamepads and controllers.

It's possible now to use gamepads and controllers as input devices. New functions have been added to work with these devices. The first one is `create_gamepad_btns_sensor`. It handles gamepad buttons. The second one is `create_gamepad_axes_sensor`. It handles gamepad axes.

- Node Logic Editor improvements.

Logic node `JS Callback` has been added. It allows to call custom JavaScript callback defined in your B4W application. Input and output parameters are supported for callbacks.

An option `Run From Script` has been added to `Entry Point` node.

Module `logic_nodes` has been added to API. It contains methods to control Node Logic Editor.

Method `append_custom_callback` has been added. It allows to register custom JavaScript callbacks to be used in `JS Callback` logic node.

Method `remove_custom_callback` has been added. It allows to remove registered custom JavaScript callback.

Method `run_entrypoint` has been added. It allows to activate Entry Point node from API.

- A new function has been added into the controls module: `create_hmd_position_sensor`.

This function allows to create a special sensor, which can track the position of an HMD device.

- The options Tilt Angle and Tilt Random are now supported for the Emitter particle systems.
- Reflection for transparent objects.

Before this release only opaque objects could be reflected. Now, transparent objects are also supported.

- Updated math modules.

Math modules `vec3`, `vec4`, `quat`, `mat3`, `mat4` are now based on `glMatrix v2.3.1`. This new version introduces `hermite`, `bezier`, `slerp`, `fromRotationTranslationScale` and `fromRotationTranslationScaleOrigin` methods.

- A new flag `Bake only deform bones` has been added to the Skeletal Animation Baker.

Previously, it was impossible to bake bones without the `deform` flag. This feature can be helpful in cases when some object is parented to the non-deforming bone.

- New GL Debug switch in the Viewer app.

Viewer application now has the GL Debug switch which allows to disable GL error checking. This increases the performance and can be useful while profiling a scene.

- Enable WebGL 2.0 for Firefox browser.

It's now possible to use experimental WebGL 2.0 context in Firefox browser.

- New `prevent_caching` engine configuration option.

This option enables/disables assets caching.

- New physics function has been added.

`apply_force_world` function applies a constant force to the object in the world space.

Changes

- Viewer's HUD Info now has detailed info on each column.

- Node Logic Editor changes.
 - An option to select between Number and String operand types has been added to the Conditional Jump node.
- The activation of the VR mode from now automatically changes the camera type to EYE for better user experience.

Fixes

- Fixed anchors behavior.
- Fixed a cubemap issue for some old NVIDIA GPUs in Firefox.
- Fixed the bug for particle systems with the “Length” vertex group specified when the emitter mesh has the “Apply Modifiers” option checked.
- Fixed the behavior of the B4W_GLOW_OUTPUT node for transparent materials.
- Fixed grass map for a single flat grass terrain object.
- Refraction vectors for stack and node materials now use correct view normal.
- Fixed God Rays Blender interface tab.
- Viewer Sky parameters updates have been fixed.
- Fixed incorrect stereo (anaglyph and HMD) rendering in specific cases.
- The Page Param node bug, which always wrote result to the variable R1, has been fixed.
- Updating variable’s scope in Node Logic Editor has been fixed.
- Fixed crash with dynamically loaded scenes while using Move Camera logic node.

v16.03

New Features

- Node Logic Editor improvements.

Now you can use global variables for sharing information between threads. This significantly expands the capabilities and allows to create more complex interactive applications.

Logic node JSON has been added. It allows to parse and encode complex JSON objects.

Logic node Get Timeline has been added. It allows to get current frame from NLA or global timeline.

Logic nodes Play Animation and Stop Animation have received environment animation mode.

- New option Update Material Animation for updating animated node shaders in viewport.

This option is useful for those who often use animated shader nodes. Just turn it on to see material animation in Blender viewport.

- Separate Project Manager server from Blender addon.

Starting from this release it's possible to run the Project Manager server as a standalone application using `project_server.py` script. This eliminates the requirement to start Blender to be able to run the Project Manager.

- Add title/description to all Blend4Web demos.

This simplifies searching our demos in Google and other search engines.

- New modules have been added.

The first new mathematical module that was added is called `math`. For more information, please take a look at [the API doc](#).

Also, an `input` module has been added. This module provides an interface for input devices: mouse — `DEVICE_MOUSE`, keyboard — `DEVICE_KEYBOARD`, touchscreen device — `DEVICE_TOUCH`, gyroscope device — `DEVICE_GYRO`, head-mounted device — `DEVICE_HMD`. For more information see API documentation on the `input` module.

- Changed policy for backward compatibility with previous Blender releases.
Starting from this release we will strive to keep the addon compatibility with previous Blender versions.
- HTML links have been supported in an anchor description.
- Objects from secondary scenes(rendered to textures) can now be controlled with logic nodes.
- The Dynamic Grass option was added to the render interface.

There are 3 options available: ON to enable dynamic grass constantly, OFF to disable dynamic grass completely and AUTO to automatically detect objects with dynamic grass.

Changes

- API changes.

The `enable_controls` and `disable_controls` of `app` add-on have been declared deprecated.

Several methods of `controls` module have been declared deprecated: `register_keyboard_events`, `register_mouse_events`, `register_wheel_events`, `register_touch_events`, `register_device_orientation`,

unregister_keyboard_events, unregister_mouse_events, unregister_wheel_events, unregister_touch_events, unregister_device_orientation.

The `create_touch_click_sensor` and `create_hmd_quat_sensor` methods have been added to the `controls` module.

The `get_hmd_device` and `reset_device` methods of `hmd` add-on have been declared deprecated.

Several methods have been added to `util` module: `deg_to_rad`, `rad_to_deg`, `quat_to_ordered_angles`.

The `get_world_by_name` method has been added to the `scenes` module.

- API for camera has been changed.

The `calc_ray()` method was changed. Now it works with parametric lines.

- Static physics behavior has been changed.

Objects, which have material with the Material -> Special: Collision property enabled, can be fully unloaded. Also, these objects can be moved, rotated, etc, as long as they are dynamic.

- Rendering optimizations.

Now we use bounding ellipsoids instead of spheres to frustum cull static objects.

- Node Logic Editor changes.

Node Send Request has been simplified. JSON parsing and encoding routine has moved to new JSON node.

- Material updates, e.g. animated or affected by dynamic lights were optimized.
- Generated water mesh has been enabled on devices without OES_depth_texture support

Fixes

- Fixed webplayer menu behavior.
- Fixed rare development server crash when the response headers contained a specific date (29Feb).
- Fixed Refractions which was set to “ON” in the absence of refractive objects.
- Fixed water material position for dynamic objects.
- Fixed crash for generated water mesh without waves.
- Fixed incorrect output for the TEXTURE node with no texture selected.

v16.02

New Features

- Extended support of Emitter type particle systems.

The support for the node materials that can be used to set particle shading parameters with sequences of basic blocks (including the Particle Info node). This function is available for the particle systems with the Billboard rendering type.

The shader for particles with the Halo type rendering has been rewritten. The support for the Rings, Lines and Star Tips parameters has been added. Maximum particle size limit (caused by the hardware limitations on some platforms) has been removed.

Maximum number of gradient control points limit in the Ramp procedural texture (used for coloring particles) has been removed.

- Extended node material support.

Two new nodes, Vector Curves and RGB Curves, have been added.

The ColorRamp node support has been added. For now, this node supports Linear and Constant type interpolation.

The Particle Info node support has been added. For now, it is fully supported by the Emitter type particle systems with the Billboard rendering type.

- Project Manager improvements.

- An option to export multiple projects into one archive.

This function makes updating the SDK and migrating projects from one workstation to another significantly easier.

- New project deploying options.

Project deployment is required for uploading finished project to the server, sending them by mail and so on. Starting with the current release, project deployment can be performed directly from the Project Manager's graphical interface. During deployment, projects are packed into a zip archive.

Also, starting with the current release, deployment is available for any type of project (including External type).

- Improvements in the Web Player HTML and Web Player JSON type projects.

Now, during the creation of these projects, you can set the Web Player application parameters such as FPS counter, automatic camera rotation, turning off social network buttons and so on.

Also, projects of these types can now be created with a bundle option which means that all application resources will be located in the same directory.

- Graphical interface improvements.

An option to add and show application icons has been added to make navigation easier and to give users a quick preview of an application in development.

An option to view project info. By clicking the [info] link located at the right side of the project name, a list of detailed information regarding the project can be accessed.

For convenience, the type of a project is now indicated by the prefix at the right side of the link: player: for Web Player HTML or Web Player JSON type projects, dev: for projects under development and build: for compiled (obfuscated) version of an application.

The elements of the Project Manager interface now have pop-up tips.

- The possibility to animate environment settings.

The possibility to animate environment parameters located in the Sky (Horizon Color, Zenith Color), Environment Lighting (Energy), and Mist (Minimum, Start, Depth, Height, Fog Color) tabs has been added. These functions are also available for the NLA animation.

The Animation tab has been added to Blender's World panel. It has Apply Default Animation and Behavior parameters.

For all exported environments, WORLD type meta-object are added to the scenes. These objects can be used to control the animation of the environment settings. You can access these objects the same way you can access any standard object in the scene.

- Camera improvements.

An option to set the limits for vertical movement of camera's pivot point has been added. This function is also available via the API by using the `target_set_pivot_limits()` and `target_get_pivot_limits()` methods.

The new `static_setup()`, `eye_setup()`, `target_setup()`, `hover_setup()` and `hover_setup_rel()` methods have been added for changing and complete setup of the camera behavior. At the same time, the `set_move_style()` has been declared deprecated.

The `target_switch_panning()` method has been added for controlling camera panning, and the `get_view_vector()` method has been added for retrieving the camera's line of sight vector.

The examples of use the camera API are now described in the corresponding chapter of the documentation. This chapter can also be accessed from the `camera.js` API module documentation page.

- Node Logic Editor improvements.

The Content-Type option has been added to the Send Request node. It can be used to reassign the title field of an HTTP request.

- A new sensor has been added to the controls module.

Gyro Quat (`create_gyro_quat_sensor` method) is the sensor for working with gyroscopes on mobile devices. It can be used to handle device rotation quaternion.

- Experimental UC Browser support has been added.
- User Manual has been reworked and expanded significantly.

Changes

- The Generate Shadows light source parameter has been renamed to Shadow and now also enables shadow rendering in the Blender Viewport.
- Design of the SDK main page has been changed.

Project list is now located in the Project Manager. Links to run frequently used applications, Viewer and Code Snippets, have been added.

- The `get_coords_x()` and `get_coords_y()` can now receive the `target_touches` parameter.

In case of multitouch, this parameter can be used to use only the touches that are inside of the current target element (the `targetTouches` parameter).

- camera API module changes.

The `has_vertical_limits()` and `has_horizontal_limits()` methods have been declared deprecated. The `has_vertical_rot_limits()` and `has_horizontal_rot_limits()` are recommended to use instead of them.

- Changes in the naming rules for the files of the projects under development.

Now, the HTML files of the projects under development (located in the `apps_dev` directory) do not include the `_dev` suffix in their names. To distinguish these projects from the compiled ones, prefixes has been added to the Project Manager.

- When you open the SDK main page, presence of the local development server is checked. If it starts from the local file system, an appropriate warning is shown.

Fixes

- Incorrect behavior of the `canvas_resolution_factor` engine parameter on Apple iOS devices has been fixed.

- Engine error that occurred in case of absence of the selected object in the Show Object and Hide Object logic nodes has been fixed.
- The incorrect coordinate rounding along the Y- and Z-axis while using variables as parameters in the Transform Object logic node has been fixed.
- Node logic blocking in case some nodes are not linked with the Entry Point node has been fixed.
- The issue with the rendering of dynamic objects the RTT-scenes has been fixed.
- Fullscreen switching issue in the Safari browser has been fixed.
- The issue with the Add-on interface (caused by Emitter type particle systems without a material assigned to the object) has been fixed.
- The Factor output of the B4W_GLOW_OUTPUT node in the materials with the Alpha Clip type transparency now works correctly.
- Dupli groups with None duplication type are not exported and not rendered.
- The `append_ray_test()` method now works correctly if its first parameter is an empty object.
- Several water settings has been optimized and fixed.
- Physical objects that have parent object will now correctly update their position in the case the physics is disabled in the scene.
- The behavior of the Camera Data and B4W Vector View nodes for reflected objects has been fixed.
- NLA animation of the RGB node in materials has been fixed.
- Now, HAIR type particle system will not be rendered if the emitter object has the Do Not Render parameter enabled.
- The work of the `auto_rotate()` method for EYE type cameras has been fixed.
- The export of Hair type particles from non-active scenes in Blender has been fixed.
- Runtime libraries for Windows have been added to fix the resource conversion error.
- Shader generation error that occurred in case there were more than 10 textures in the material has been fixed.

Known Issues

Starting with this release, the list of all known problems and possible solutions is located in the [dedicated chapter](#).

v16.01

New Features

- Line rendering.

Procedurally generated line rendering is now supported. Special type of object, activated by the Line Renderer option in blender, is provided for it, as well as several API functions: `draw_line`, `get_line_params` and `set_line_params`.
- New logic editor nodes.
 - The Transform Object node can be used to move an object in world, parent or local coordinate space.
 - The String Operation node can be used to perform operations with string constants and variables, like the Math Operation node.
- Simplified SDK installation.

Now, to [install SDK](#), you just need to specify the path to SDK in the File->Scripts section of the User Preferences panel.
- WebPlayer application improvements.

An option to turn off social network buttons located in the bottom-right corner of a loaded scene. To do this, you need to specify the `no_social` attribute before the application starts.

Escape characters in the Web Player's address bar are now processed correctly.
- Rendering quality improvements for Head-Mounted Displays (HMD).

Means to correct distortion and disable chromatic aberration while using Head-Mounted Displays have been added. Use `set_hmd_params()` function to set up these parameters.
- Two new sensors have been added to the `controls` module.

Touch Rotate (`create_touch_rotate_sensor()` method) sensor can be used to process rotation using two fingers on touch devices.

Callback (`create_callback_sensor()` method) is a sensor whose value is defined by a callback function called every frame.
- Resource converter utility improvements.

The `-verbose` and `-jobs` parameters have been added to the `resource converter utility` `converter.py`, which can be used to output detailed information and to set maximum number of the parallel processes during conversion, respectively.
- Several API methods have been added.

`is_line()` checks whether the object has LINE type.

`is_idle()` checks whether the scene loader has finished all planned tasks.

`hover_switch_horiz_rotation()` can be used to enable and disable horizontal rotation of HOVER type cameras.

- Node tree refresh algorithm has been optimized.

Changes

- Using cubic reflections has been simplified.

Cubic reflection is now rendered from the object's geometric center and not from its Blender origin point. Also, flat reflection is no longer rendered when rendering cubic reflection, which caused artifacts before.

- Changes in API.

The following methods have been added to the camera module: `get_vertical_axis()`, `set_vertical_axis()`. “EYE” type camera rotation to a “phi” angle is now performed along the vector returned by the `get_vertical_axis()` function.

The `correct_up()` method now accepts the strict flag as a parameter. This parameter can be used to align camera codirectionally with the `y_axis` vector and not simply parallel to it.

`eye_get_vertical_limits`, `eye_get_horizontal_limits`, `target_get_vertical_limits` and `target_get_horizontal_limits` methods can now return limits set in both world and local coordinate spaces. This can be defined by the local parameter.

- Logic node changes.

The nodes have been separated into categories.

- An option to select between Number and String variable types has been added to the Variable Store node.
- An option to use string variables to store the entire body of the server request and response has been added to the Send Request node.

- Deprecated Mass Reexporter tool has been removed.

The Mass Reexporter tool panel has been removed, as automatic scene reexport function is already present in the Project Manager (re-export scenes operation).

- Deprecated functionality.

The following methods: `mouse.enable_mouse_hover_glow()`, `mouse.disable_mouse_hover_glow()`, `anim.get_actions()`, `anim.get_current_action()`, `anim.set_current_frame_float()`, `anim.get_current_frame_float()`, `anim.get_frame_range()`,

`anim.cyclic()`, `anim.is_cyclic()`, `anim.update_object_animation()`,
`controls.remove_sensor_manifolds()`, `main.redraw()`,
`scenes.set_glow_intensity()`, `scenes.get_glow_intensity()`,
`scenes.apply_glow_anim()`, `scenes.apply_glow_anim_def()`,
`scenes.clear_glow_anim()`, `scenes.set_glow_color()`,
`scenes.get_glow_color()`, `sfx.speaker_play()`, `sfx.speaker_stop()`,
`sfx.speaker_playback_rate()`, `sfx.get_speakers()`,
`trans.set_rotation_quat()`, `trans.set_rotation_quat_v()`,
`trans.get_rotation_quat()` have been removed.

The `set_camera_move_style()` method and `SPACE_WORLD` and `SPACE_LOCAL` constants have been declared deprecated.

The `has_vertical_limits()` and `has_horizontal_limits()` methods have also been declared deprecated. The new methods `has_vertical_rot_limits()`, `has_horizontal_rot_limits()`, `has_vertical_trans_limits()` and `has_horizontal_trans_limits()` have been added to replace them.

Fixes

- Positioning and flickering of the particle systems with world coordinates have been fixed.
- Front Facing for cubic reflections has been fixed.
- Stereo mode rendering errors have been fixed.
- Incorrect positioning in the child object animation has been fixed.
- Scene loading freeze in the Safari browser while using .ogg audio files with Background Music type speakers has been fixed.
- The Move Camera node bug, which disabled Duration parameter after .blend file reopening, has been fixed.
- System crash while exporting particle system with complex mesh emitter has been fixed.
- The Switch Select node error that led to incorrect switches has been fixed.
- Several camera limit rendering issues in the Blender viewport have been fixed.
- The issue with exporting materials attached directly to the object and not to the mesh has been fixed.
- The RenderCallback function set by the `set_render_callback()` method has been fixed.

The RenderCallback function are now called right before rendering the current frame, so the scene and objects are up-to-date.

- Issues with rendering shadows from multiple light sources of different types (such as POINT and SUN) have been fixed.

- Environment texture rendering while using the texture as the world map and in a stock material at the same time has been corrected.

Known Issues

- Problems with updating of the add-on.

It's strongly advised to restart Blender after installing a newer version of Addon/SDK.

- NVIDIA 331 driver in Linux can cause WebGL errors.

- Changed texture filtering on some platforms.

An incorrect texture filtering was disabled on iPad and Internet Explorer for materials with Alpha Clip type of transparency.

- Some devices with Mail GPU require manual WebGL activation in browser settings.
- For the local development server to work on Apple OS X and Blender 2.76, you may need to install [Python 3.4](#). This is due to a bug in Blender <https://developer.blender.org/T46623>. This bug has been fixed in Blender 2.76b, so updating it is advised.
- Skeletal animation can work incorrectly while using Nouveau drivers.

v15.12

New Features

- Support for shadows from multiple sources.

Support for shadows from multiple sources has been added. This feature can be used to greatly improve the realism of scenes lit by multiple light sources. Its functions and limitations are described in the documentation.

- Experimental support of the HMD (Head-mounted display).

Experimental support for the head-mounted displays with the WebVR API has been added. For now, using this technology requires a web browser with WebVR support and an Oculus Rift device. More details in the documentation.

- Aligning objects with the camera in the viewport.

The new Viewport Alignment option can be used to attach objects to the camera in the viewport. This feature can be used to create UI elements attached to the center, edge or corner of the screen. Unlike parent-child alignment, the position of an aligned object will automatically change when screen resolution or aspect ratio is changed.

Functionality of the `append_stiff_viewport()` method used for Stiff Viewport constraint has also been expanded.

- An option to show camera limits in Blender viewport.

To make the task of setting camera limits up simpler, the Display limits in viewport option has been added to the Data panel of the camera object. When enabled, it will show the limits right in the Blender viewport. Default settings of the limits has also been changed.

- Anaglyph rendering improvements.

New anaglyph rendering algorithm has been implemented. It has better color rendering. Also, plane of convergence of the left and right image for a TARGET type camera will now be calculated automatically based on a point around which the camera is rotated.

- New logic editor nodes.

- Move To: Can be used to move an object to another object.
- Console Print: Can be used to print various information (including variables values) to the web browser console.

- A Stereo View button has been added to the Web Player.

Can be used to switch to the HMD mode (if the web browser supports this technology) or to anaglyph mode.

- New methods for working with objects.

The new methods `get_selectable_objects` and `get_outlining_objects` have been added to make working with objects simpler.

- Dynamic objects can now be deleted.

Before, only objects copied by the user could have been deleted.

- The option to choose an annotation of an object based on the canvas coordinates has been added.

The `pick_object` function now returns the object's annotation, if the object is present in the sent coordinates.

- A new method to initialize media resources for mobile devices has been added.

Before, forced canvas block was used as a workaround of the mobile browsers restriction of loading media resources (video and audio), and a user had to click a popup icon to start the application. Now, you can manually initialize media resources by using `activate_media` method. This function can be useful if the application already have elements that require user input, such as Start button, sound mute button and so on.

- New sensor manifold type - CT_POSITIVE

The new CT_POSITIVE sensor manifold type activates event handler if the logic function result isn't zero.

Changes

- The Fast Preview button has been duplicated in the UI low panel.

This button can be used to preview the scene without switching to the Development Server scene settings panel.

- Canvas texture API has been changed.

Now working with a canvas texture is simpler. New methods `get_canvas_ctx` and `update_canvas_ctx` have been added. The `get_canvas_texture_context` and `update_canvas_texture_context` methods have been declared deprecated and are not recommended to use. The Source ID interface field for the canvas texture has also been removed.

- Wrong behavior of the shadows from Alpha Clip materials has been fixed.

Incorrect behavior of the shadows casted by the objects with Alpha Clip materials without transparency value has been fixed.

Fixes

- Incorrect physics of a copied object has been fixed.

Incorrect physics behavior of a copied object (which occurred if the object was moved before being added to the scene) has been fixed.

- Incorrect fog behavior if a water plane was added to the scene has been fixed.
- Misplacing of the Blender interface panels in basic scene for a new project has been fixed.
- Cubemap rendering issue on the NVIDIA GeForce 200 series GPU has been fixed.
- Engine workflow on iPhone (4, 4S and 5), iPad (2nd, 3rd and 4th generations) and iPad Mini (1st and 2nd generations) has been improved.
- Incorrect rendering of the shadows casted by billboard objects has been fixed.

Known Issues

- Problems with updating of the add-on.

It's strongly advised to restart Blender after installing a newer version of Addon/SDK.

- NVIDIA 331 driver in Linux can cause WebGL errors.
- Changed texture filtering on some platforms.

An incorrect texture filtering was disabled on iPad and Internet Explorer for materials with Alpha Clip type of transparency.

- Incorrect Depth texture behavior on Windows OS.

Depth textures are unstable on Google Chrome 32 bit on Windows. This problem is fixed in the beta version of the browser.

- Some devices with Mail GPU require manual WebGL activation in browser settings.
- For the local development server to work on Apple OS X and Blender 2.76, you may need to install Python 3.4. This is due to a bug in Blender <https://developer.blender.org/T46623>. This bug has been fixed in Blender 2.76b, so updating it is advised.

v15.11

New Features

- Initial support of WebGL 2.0 technology.

New API features are automatically provided with compatible browser and hardware

- New Logic Editor Features:

- Camera animation.

The option to smoothly change camera's position has been added to the Move Camera node.

- New: behavior animation.

The Play Animation node can now add Behavior animation to an object.

- New Stop Animation node

Can be used to stop object's animation

- New Stop Sound node

Can be used to stop a speaker's sound playback

- New Stop Timeline node

Can be used to stop NLA animation

- MSAA support

This antialiasing algorithm is used in systems compatible with WebGL 2.0. For now, it allows the highest image quality on the ULTRA settings by using 16x MSAA algorithm.

- Viewer interface improvements.

A Home button has been added. When pushed, it will open the basic scene specified in the application's URL parameters, or, if there isn't any, a placeholder scene with Blend4Web cubic logo will be opened.

The Reset button has been moved to the right side of the application's upper panel and will now reset not only internal but also URL parameters (such as loaded basic scene) of the Viewer.

The Auto View Mode button used for automatic walkthrough of all scenes has been moved to the Tools & Debug panel.

- High precision rendering performance measure mechanic.

Using the WebGL extension EXT_disjoint_timer_query allows high precision timers to be created, which can be used to measure the performance of GPU operations. With the activation of the debug mode in the Viewer ('HUD Info' option in the Tools & Debug panel), the time (in milliseconds) the GPU needs to render part of the scene (subscene) is shown in the last column of the debug information.

- Web player improvements.

Now, the cursor changes appearance while the camera is moving.

- There are numerous additions and improvements in the documentation.

Changes

- The Copy Cursor Location button has been renamed to Look At Cursor, and its behavior has also been changed.

Now, by pressing this button, you can rotate the camera to the pointed direction.

- Changes API.

For compatibility with third-party applications, `:b4wref:transform.set_matrix`, `get_matrix`, `set_matrix_rel` and `get_matrix_rel` methods have been added, but for optimization purposes, using `set_tsr`, `get_tsr`, `set_tsr_rel` and `get_tsr_rel` methods is recommended.

Now the `hide_object` and `show_object` functions will, by default, change the visibility of an object and all its children. To change only the current object's visibility, `ignore_children` parameter with true value should be sent.

The following method has been declared deprecated and will be removed in further releases: `create_sep()` (`tsr` module). `tsr` module's `set_sep()` method should be used instead of it.

`camera` module API has been changed considerably. Deprecated methods `is_camera()`, `rotate_pivot()`, `rotate_hover_cam()`, `get_hover_cam_angle()`, `set_hover_cam_angle()`, `translate_hover_cam_v()`, `set_eye_params()`, `rotate()`, `get_angles()` have been removed. The following methods have also been

declared deprecated: `set_look_at`, `rotate_eye_camera`, `rotate_target_camera`, `set_trans_pivot`, `zoom_object`, `set_pivot`, `rotate_hover_camera`, `get_hover_cam_pivot`, `get_eye`, `get_pivot`, `hover_cam_set_translation`, `set_hover_pivot`, `get_hover_angle_limits`, `get_cam_dist_limits`, `apply_vertical_limits`, `clear_hover_angle_limits`, `apply_hover_angle_limits`, `apply_distance_limits`, `clear_distance_limits`, `get_vertical_limits`, `apply_horizontal_limits`, `get_horizontal_limits`, `clear_vertical_limits`, `clear_horizontal_limits`. In their place, we recommend using methods starting with camera type prefix: `target_...`, `eye_...`, `static_...`, `hover_....`. Methods `set_velocity_params` and `get_velocity_params` have also been declared deprecated, and instead of them, using `set_velocities` and `b4wref:camera.get_velocities`, respectively, is recommended

For working with the distance between the current TARGET or HOVER camera and the pivot point, `target_get_distance`, `target_set_distance` and `hover_get_distance` methods have been added

- In the Viewer, Play All and Stop All buttons have been added for playing and stopping all animations, respectively.
- Logic editor changes:
 - Select node and Select & Play Animation and Select & Play Timeline nodes based on it, have been declared deprecated. As a replacement, using the combination of Switch Select, Play Animation and Play Timeline nodes is recommended.
 - SDK examples, which were using aforementioned deprecated nodes, have been updated.
 - For the logic nodes behavior transparency, Cyclic NLA option has been switched off.
- Behavior of the HOVER camera in the absence of limits has been changed.
 - HOVER camera will now be hanging over pivot point at a fixed distance and fixed angle.
- Glow effect changes.

Light sources will now influence glow effect, if Material or Extended Material nodes are present.

Fixes

- Project Manager errors were fixed.

Links to the developer's applications included in the project are now showing in the list.

Formatting in the compiled applications' HTML files has been fixed.
- Skinning on the mobile platforms has been improved.

- Addon's translation files connection error has been fixed.

An error that could have happened on some devices because of system's inability to process addon's translation files, has been fixed.
- Particle objects shadow casting improvements.

Fix behavior of the shadows casted by "Hair" particles (used as billboards) while changing the size of the main canvas.
- Engine workflow on the Mali 400 series GPU has been fixed.
- Flat reflections and fog for double_sided_lighting materials have been fixed.
- Local Storage limited quota on the Safari browser in the incognito mode no longer causes error.
- Functionality of the Render Glow Over Transparent Objects' option in the Blender's [glow settings](#) has been restored.
- Several video texture playback issues have been fixed.

Known Issues

- Problems with updating of the add-on.

It's strongly advised to restart Blender after installing a newer version of Addon/SDK.
- NVIDIA 331 driver in Linux can cause WebGL errors.
- Changed texture filtering on some platforms.

An incorrect texture filtering was disabled on iPad and Internet Explorer for materials with Alpha Clip type of transparency.
- Incorrect Depth texture behavior on Windows OS.

Depth textures are unstable on Google Chrome 32 bit on Windows. This problem is fixed in the beta version of the browser.
- Some devices with Mali GPU require manual WebGL activation in browser settings.
- You may require to install [Python 3.4](#) on the systems with Apple OS X and Blender 2.76. This issue is connected with Blender bug <https://developer.blender.org/T46623>.

v15.10

New Features

- New Project Manager features.

It is now possible to export/import projects. This simplifies updating projects to newer versions of the SDK and also allows users to run and debug projects on various platforms. You can also share your projects with other developers in an efficient way.

You can now create WebPlayer JSON and WebPlayer HTML projects. Such options allow you to create projects specifically for WebPlayer app which do not require writing any additional code.

To simplify navigation between your projects, two features have been added. The first one is the possibility to sort projects by name, and the second one is the possibility to show/hide SDK's stock projects.

It is now possible to remove projects from the SDK.

Improved Windows support. We now provide a Windows version of Java with the SDK, so you are no longer required to install any additional dependencies in order to build projects.

- Anti-aliasing improvements.

A new option AA Quality has been added to the Render > Anti-Aliasing panel in Blender. This option allows you to select quality level of anti-aliasing. Also, the anti-aliasing algorithm is now based on FXAA 3.11, which increases both quality and performance especially for HIGH and ULTRA quality profiles.

- Node Logic Editor improvements.

- Move Camera node.

Allows you to specify the camera's location and target.

- Play Sound node.

Allows you to play back sound from a selected speaker.

- Switch Select node.

This node works the same way as Select but allows you to select multiple objects in more flexible and convenient way.

- Support for random numbers generation has been added to the Math Operation node.
 - Support for POST requests in the Send Request node.

- Repeat mapping type for non-power-of-two textures is now supported.

Repeat mapping type is now supported for non-power-of-two textures (i.e. textures whose dimensions are not 256, 512, 1024, etc). Also the mipmapping technique (trilinear filtering) is also supported for such textures.

- Automatic rescaling of textures is now performed if their dimensions exceed platform limits.

In the cases when texture dimensions exceed platform limits, textures will be automatically downsampled. The only exception is compressed textures.

- New button Fast Preview to perform fast scene previews.

The button is located on the Render > Development Server panel.

- Support for Intensity and Color animation of lamp objects.

It is now possible to animate intensity and color of lamp objects, both when using conventional and NLA animation.

Changes

- Project Manager interface has been improved.

Improved UI, added Development Server > Project Manager button to run the Project Manager in the default browser.

- Speakers functioning has been improved.

The `is_play()` method now correctly notifies about finishing sound playback, with a minimal delay.

- Some new export warnings have been added.

Upon exporting some objects, their type will be changed to EMPTY in case of empty geometry or in the case when the sound file for the SPEAKER object does not exist. Messages on such facts are now displayed in the browser console.

- Displaying the object selector for logic nodes has been improved.

- Gamma correction behavior in node materials has been changed.

Gamma correction in node materials is now performed differently because of changes in Blender 2.76.

- Changes API.

The `resize` method is now deprecated and will be deleted from the `main` module. The `resize` method from the `container` module should be used instead.

Fixes

- Project Manager errors were fixed.
- Stability on mobile devices has been improved for scenes with too many lamps.
- Stability on Unix systems has been increased.

Stability on Unix-systems using AMD GPUs and open source drivers has been increased.

- An error related to annotations being added to a scene has been fixed.

Fixed an error which occurred when annotations without the Object -> Meta Tags property were added to the scene.
- Fixed an error with incorrect audio playback during browser tabs switch.
- Several video texture errors have been fixed.
- Fixed an error occurred when an empty node group was used inside a node material.
- Fixed an error with a particle system which use an object with LOD as a particle.
- The Delay node error when using a variable as a parameter has been fixed in the logic editor.
- Fixed an error in the logic editor which occurred upon deleting an Entry Point node.
- Fixed an error in the logic editor with duplicated variables in the dropdown list.
- Fixed an error in scenes which use both the Play Animation logic node and the switched off NLA flag.
- The behavior of the Math->Power and Gamma shader nodes has been fixed and is now consistent with Blender.

Known Issues

- Problems with updating of the add-on.

It's strongly advised to restart Blender after installing a newer version of Addon/SDK.
- NVIDIA 331 driver in Linux can cause WebGL errors.
- Changed texture filtering on some platforms.

An incorrect texture filtering was disabled on iPad and Internet Explorer for materials with Alpha Clip type of transparency.
- Incorrect Depth texture behavior on Windows OS.

Depth textures are unstable on Google Chrome 32 bit on Windows. This problem is fixed in the beta version of the browser.
- Some devices with Mail GPU require manual WebGL activation in browser settings.
- You may require to install [Python 3.4](#) on the systems with Apple OS X and Blender 2.76. This issue is connected with Blender bug <https://developer.blender.org/T46623>.

v15.09

New Features

- Project Management.

A new project management system allows one to:

- show the list and info about existing projects
- execute apps, load scenes in the Viewer app, load scene files in Blender.
- create and configure new applications, optionally using prefabricated app/scene starter files to simplify creating new applications
- build applications, create versions suitable for deployment on a server
- convert application resources (textures, sounds and video) to alternative formats
- automatically reexport all application scenes including json and html files

There are two methods to manage projects: by using `project.py` utility which has a simple self-documented command line interface, or using a GUI web application, which is run on the developer server. This server does not require any additional configuration and as before is executed by the Open SDK button in Blender.

The project management system works on all operating systems, however, some operations may require additional dependencies. To find out more about the configuration of this system check out the following [topic](#) in documentation.

- Controlling armature bones.

Enhanced features to control armature objects.

Added a new `armature` module which includes methods to get or assign bone positions both in armature coordinate space (`get_bone_tsr()`, `set_bone_tsr()`) and in bone-relative coordinate space (`get_bone_tsr_rel()`, `set_bone_tsr_rel()`). By using these methods, it's possible to program armature behavior. For example, animate interactive characters or create sophisticated armature-based objects with multiple moving parts.

Support for `COPY_TRANSFORMS` constraints on bones. This allows bones to follow movements of any objects located on the scene, e.g create physically simulated "Ragdoll" objects.

- New logic editor nodes.

- Play Animation: play animation of an object.

- Send Request: send HTTP GET request to a server and parse its response.
 - Inherit Material: copy material properties from one object to another.
 - Set Shader Node Param: change outputs of Value and RGB shader nodes.
 - Delay: delay program execution for a given amount of time.
 - Apply Shape Key: change shape key value for an object.
 - Outline: control object outlining effect.
- New Allow NLA settings for video textures, which enables/disables NLA-animation for a given texture.
 - New features in the material module.

New APIs added `set_specular_color_factor` and `get_specular_color_factor`.
 - New features on the SDK index web page.

Index page of the SDK now contains a Tools section, which in turn includes Project Manager and WebGL Report tools.
 - Web player improvements.

A new optional alpha setting has been added, which is used to set rendering canvas transparency. The [Outline on Select](#) feature is now supported.
 - New app compilation type update has been added to `project.py` utility.

This type of app compilation allows one to update the engine inside a project directory. The app itself is not compiled with this option. The feature is useful for basic applications and tutorials.
 - New features in the app module.

A new param `disable_zoom` has been added to the `enable_camera_controls()` method. This parameter disables zoom movements of the camera.

A new method `queue_animate()` has been added to the `app` module.
 - New features in the scenes module.

A new method `has_picking_subs()` has been added to the scenes module.
 - New features in the debug module.

A new method `fake_load()` has been added to the `debug` module.

Changes

- Add-ons (such as `app`, `mouse` and others) are now a part of the compiled engine version:
 - `b4w.min.js` - advanced optimization (`b4w.full.min.js` previously)

- b4w.simple.min.js - simple optimization
- b4w.whitespace.min.js - optimization of whitespaces used in the code

The correct version is chosen according to the application compilation settings.

- Node Logic Editor improvements.
 - Play -> Play Timeline; Select & Jump -> Select, Register Store -> Variable Store nodes were renamed.
 - Now it is possible to create user-defined variables apart from register-variables.
- The rarely used combine engine building method was removed.
- Documentation for the [resource converter](#) was revised.
- Documentation for the [addon translator](#) was revised
- The export of UV-layers and vertex colors was changed.

Now the behavior of UV-layers and vertex colors resembles that in Blender even more.

- Support for NLA-animation and video-textures was extended. They act similar to those in Blender.
- A message about the lack of animation channels was added to the [export](#) errors.
- A non-critical export error on the selection of unsupported Render Type in particle systems was added.
- The Specular Color Factor property is now being inherited during material inheritance ([inherit_material](#) API method).
- Changes API.

The following methods are marked as deprecated and will be removed in future releases: `is_camera`, `is_mesh`, `is_armature`.

The following methods of the `objects` module should be used instead: `is_camera`, `is_mesh`, `is_armature`

Also, there are the following new methods: `is_speaker`, `is_lamp` and `is_empty`.

The `get_object_by_dupli_name_list` method now returns null, if the `name_list` parameter is given incorrectly. The `get_object_name_hierarchy` method for receiving the full list of names with respect to object duplication was added. The value returned by this method is a valid input for the `get_object_by_dupli_name_list` function.

The `wireframe_mode` parameter in the `set_debug_params` method now has a value of one of the following constants: `WM_NONE`, `WM_OPAQUE_WIREFRAME`, `WM_TRANSPARENT_WIREFRAME`, `WM_FRONT_BACK_VIEW`, `WM_DEBUG_SPHERES`.

- NLA Animation Behavior for dynamically loaded scenes changes.

If there are objects with NLA-animation in such scenes, they are now influenced by the NLA settings of the main scene. Previously, such animation was not supported.

- The refactoring of the objects' internal structure was continued.
- Export error was added.

Now when an object with a type other than "Mesh" is used as a "Hair" particle, a non-critical `export` error will occur.

- Licensing information was added to the distribution sources.
- Now texture slots with Environment Maps containing video textures are not exported.

A `non-critical export` error, stating that a video cannot be used as an Environment Map, was added.

Fixes

- Render Above All option now works correctly with node materials.
- Fixed `remove_object()` function.
- "Hair" particles have become more stable.
Fixed the bug that appeared while using an object with the "Hair" particle modifier.
- Improved Windows Phone support.
- Fixed the bug that appeared while copying physical objects.
- Fixed the bug with Orco vector output when an object has zero scale in one or several axes.
- Fixed the bug in particle emitters: it appeared when an object with physics settings was chosen as a particle.
- Fixed the bug in NLA animation: it could not start from a frame other than 0 before.
- Fixed Lamp Data behavior: previously the information about light sources was not always refreshed during scene loading.
- Fixed the bug in calculations of Normal vector output in node Geometry on the back side of a polygon.

- Fixed the bug of Orco vector output in node Geometry that appeared if object was translated relatively to its origin in Blender.
- Fixed calculation of the last frame of NLA animation for video textures.
- Fixed the engine bug related to different setting combinations of objects' selection and outlining.
- Wind bending effect has become more stable.
- Fixed bugs in Alpha Clip materials rendering.
- Fixed specular texture reproduction during material inheritance.
- Fixed the bug with light sources in apps with multiple scenes.

Known Issues

- Problems with updating of the add-on.
It's strongly advised to restart Blender after installing a newer version of Addon/SDK.
- NVIDIA 331 driver in Linux can cause WebGL errors.
- Changed texture filtering on some platforms.
An incorrect texture filtering was disabled on iPad and Internet Explorer for materials with Alpha Clip type of transparency.
- Some devices with Mail GPU require manual WebGL activation in browser settings.

v15.08

New Features

- Node-based logic editor.
This new logic editor allows to add interactivity to your apps more easily by using a tree of connected nodes. The NLA Script tool which was previously used for this purpose has been removed, the old scripts are automatically converted to the newer format during blend file loading.
- Support for various shading models inside MATERIAL and MATERIAL_EXT nodes.
Now the diffuse and specular shading models are selected based on materials inside such nodes. This is different from the previous behavior when the shading was the same for all nodes and was assigned by node material itself. This feature allows mixing different basic materials (non-textured) inside node-based ones.
- Improvements in transformation API.

New `transform` methods to perform relative objects transformations: `set_translation_rel()`, `set_translation_rel_v()`, `get_translation_rel()`, `set_rotation_rel()`, `set_rotation_rel_v()`, `get_rotation_rel()`, `set_scale_rel()`, `get_scale_rel()`, `set_tsr_rel()`, `get_tsr_rel()`.

Methods without `_rel` suffix now always perform transformations in world coordinate space, even if they are children of other objects.

Also new methods to perform local rotations `rotate_x_local()`, `rotate_y_local()`, `rotate_z_local()` have been added to `transform` module.

- Improved support for OS Windows.

Support for OS Windows in `resource converter`. Now Windows users can create cross-browser applications which use media resources in different formats. Also the `application builder` app can also be run in Windows.

- Support for the Microsoft Edge browser.
- Support for physics simulation in the main (non-worker) browser process.

This feature is useful for eliminating delays in physics calculations in some mobile browsers. Earlier, all simulations took place in separate Worker threads. The feature is controlled by `physics_use_worker` parameter of the engine's initialization.

- Support for bounding box recalculation.

A new method `update_boundings()` has been added to the `objects` module. This method performs recalculations of the object's boundings such as box, sphere, ellipsoid, cylinder, cone and capsule.

- Documentation improvements.

A new theme has been selected for the user manual. This theme improves documentation readability on displays with different screen resolutions.

Greatly improved and extended documentation for `application developers`.

- New APIs in `camera_anim` module.

New methods `stop_cam_moving()` and `stop_cam_rotating()` in the `b4wmod:camera_anim` module.

- Binary compatibility checks.

Now the version checks between `.bin` and `.json` files and the current engine version are performed while scenes load.

Changes

- Changes in node material editor.

Custom shader nodes have been moved from the Group menu to the Blend4Web menu. Also, an error with duplicated menu entries has been fixed.

- Refactoring of object structure.

Complete refactoring of an object structure has been initiated. Object structures now have strong typing and include less redundant data. This improves overall engine performance.

- Changes API.

The following methods are now deprecated and will be removed in future engine releases: `get_object_dg_parent()` (`scenes` module), `get_parent()` (`constraints` module). Instead, it is recommended to use `get_parent()` and `get_dg_parent()` from the `objects` module.

- Improved interaction between the development server and multiple Blender instances.

Fixes

- Fixed a compilation error in the `project.py` utility when a project is compiled to some external directory.
- Fixed a web server error which could arise due corrupted Windows registry.
- Fixed an export freeze for some corrupted `.blend` files.
- Fixed incorrect non-uniform scale warnings upon exporting Metaball objects.
- Fixed an error with Auto View mode in the Viewer app if were an empty scene category.
- Fixed rendering of dynamically loaded materials if they have the same name.
- Fixed an export error for scenes using Copy Transforms constraints.
- Fixed an error with rendering depth textures in the Microsoft Edge browser.
- Fixed a bug with touch events in the Microsoft Edge browser.

Known Issues

- In the logic editor, some of looped links are highlighted in red. This issue has only cosmetic effect and can be safely ignored.

v15.07

New Features

- Support for Alpha Clip transparency for node materials.

It is now possible to specify a transparency mask for Alpha Clip materials using node logic.

- Support for soft particles.

The new property Soft Particles has been added to the Render panel of the Emitter particle system. When activated, this property renders smooth edges near opaque objects located on the scene. This effect is available only for materials with Alpha Sort, Alpha Blend or Add type of transparency.

- New preprocessor for shader instructions.

This new preprocessor has more straightforward architecture and works faster, which, in turn, greatly reduces overall scene loading time, especially in cases with many different shaders.

- Improved Viewer app performance.

Performance problems caused by event handling in jQuery Mobile library have been fixed.

- Zero level property for HOVER cameras.

This property represents a Z coordinate of the reference plane in which the camera's pivot point is located.

- New sensor manifold type: CT_CHANGE.

Along with CT_CONTINOUS, CT_TRIGGER, CT_SHOT and CT_LEVEL manifold types the new CT_CHANGE type can now be used. This type allows to execute a manifold's callback right after any of the sensors' values in the manifold has been changed.

- New APIs in camera_anim module.

New API methods such as `move_camera_to_point()`, `rotate_camera()`, `is_moving()` and `is_rotating()` have been added to module `camera_anim`.

- A new option to assign materials to objects.

Until now it has not been possible to assign materials on objects using Object property. Now, this feature is supported.

- Support for external requests to the local development server.

This feature is enabled by the Enable external requests property in addon preferences.

- New method `clear_animation()` in the `time` module.

- New camera APIs.

APIs of camera module has been extended by `get_vertical_limits()` and `has_vertical_limits()` methods which are used to get or check the camera's vertical rotation and translation limits.

- Documentation improvements.

New user manual design optimized for devices of all kind.

The differences between coordinate systems used in Blender and Blend4Web are now specified in documentation.

- Support for addon i18n.

This feature allows translation of addon interface to different languages. The translation into Russian is almost complete.

Changes

- Various usability improvements in the addon interface.
- New `force_container_ratio` option in the `init()` method.
- Simplified usage of water caustics.

Water caustics effect is now activated by the Caustics option located on the material's Water panel.

- New way of evaluating mouse movement sensors.

Now all elements inside the Canvas Container element (e.g Anchors) do not lock mouse events, thus allowing us to prevent glitches during camera movement.

- Option to change quality of reflections.

Now it is possible to choose one of three predefined reflection quality modes: LOW, MEDIUM or HIGH.

- New properties --project and --app in the project.py utility.

- New animation APIs.

New methods to simplify frame setting: `set_first_frame()`/`set_last_frame()`.

- New behavior of plane reflections.

It is now possible to omit specifying the reflection normal by leaving the Reflection Plane option empty. In this case, local Z axis will be used instead.

- New SSAO behavior.

Now there is no need to have shadow casters in the scene to use the SSAO effect.

- New texture anisotropic filtering settings.

There is a new Default value for anisotropic filtering option assigned to textures. By selecting this value you can use the anisotropic filtering specified on the scene. If you need to disable anisotropic filtering on all textures, again, use the setting from the scene.

- New specular alpha shading behavior to match Blender's.
- The physics mask/group, assigned for collision materials, has been extended from 8 to 16 bits.
- New asynchronous shader loader for developer version of Blend4Web.
- Elimination of shaders module.

The methods of this module is now available from the `debug` module.

- New script for batch processing exported json/html files.

The scripts `reexporter.py` and `resaver.py` have been combined into `process_blend.py`, which has options for processing exported json/html files.

- Minor refactoring and improvements in API documentation.

Fixes

- Fixed a bug with incorrect Canvas element size appeared on the engine's startup.
- Fixed a lighting bug on objects with the negative Scale option.

This condition is properly handled and reported to the user (in the browser console).

- Fixed a bug with object's vertex animation.
- Fixed a bug with animation played in reverse.
- Fixed an error with incorrect state of Collision and Ray sensors which appeared after deleting physics objects.
- Environment textures with dimensions exceeding the supported ones are now processed correctly.

Now, such textures are not being turned off but rendered in reduced scale instead. Dimensions are reduced synthetically for NVIDIA GeForce 8000 GPUs on Windows Chrome.

- Fixed an issue with incorrectly reported error which appeared while loading scenes with missing camera/world.
- Fixed a Blender startup error with the world missing from the scene.
- Fixed a bug in the Webplayer app when the sound button was missing in some scenes.
- Fixed a bug in the scenes when motion blur is the only effect to appear.

- Fixed the `get_material_extended_params()` method in the `material` module.
- Fixed Firefox Mobile crashes when using shadows. Improved overall stability for this browser.
- Improved reexporter stability.

Known Issues

- Problems with updating of the add-on.

It's strongly advised to restart Blender after installing a newer version of Addon/SDK.
- NVIDIA 331 driver in Linux can cause WebGL errors.
- There is a [bug](#) with video textures on Chrome 43 for Android.

Please update your Chrome browser to Beta or wait until the next Chrome update.
- Fixed issues with the Background Music speakers in the scenes exported to html.

Currently, there is a bug in Google Chrome ([Issue 511251](#)), related to an error with audio file origin. Currently, this issue is partially resolved by our workaround with forced `crossOrigin` attribute on audio sources.
- Changed texture filtering on some platforms.

An incorrect texture filtering was disabled on iPad and Internet Explorer for materials with Alpha Clip type of transparency.

v15.06

New Features

- New add-on user interface.

Add-on interface has been redesigned. It is now activated by the new rendering profile, Blend4Web, which only contains panels and options explicitly supported by the engine. To simplify navigation, the old multi-line Blend4Web panels have been re-grouped into the smaller ones, based on functionality.

Also, there is a new feature to automatically assign graphic effects required for a scene. In particular, shadows, refraction, Glow and Outline effects acquired the new AUTO property which automatically activates them if some objects or materials located on the scene require such effects.
- New normal editor.

The normal editor has been substantially upgraded. Now it is fully compatible with the native Blender datablock used to store normals. This new editor has more efficient UI and also allows to edit split normals.

- Support for new material nodes.

Orco and Local outputs of Geometry node are now supported. There are also some preliminary steps to support RGB Curves, Vector Curves, ColorRamp and Cycles nodes.

- New shading models.

Minnaert/Toon diffuse shaders and Blinn specular shader have been implemented. Thus, starting from this release the engine supports all shading models of Blender.

- Support for Blender 2.75

- Multiple physics improvements.

Code for collision detection has been rewritten. Now it is possible to determine colliding objects, and also the coordinates and the normal at the collision point.

Improved ray casting API. In particular, one can specify an option to perform an automatic cleanup of the ray test object and also another option to cast a ray through multiple objects. As in the case of collision detection, this new API allows to determine the target object and the position/normal of the hit point. There is also a new possibility to cast rays from point to point in global space, without requirement to specify a source object.

Extended possibilities of Collision and Ray sensors.

Support for deleting physics objects and automatic recalculation of collision/ray tests after physics objects have been added/removed.

- A new tool for reexporting multiple scenes.

A new Mass Reexporter tool has been added to addon. This tool allows to automatically reexport all scenes from the specified list of directories.

- Possibility to check for updates.

You can now enable a Check for Updates on Startup option in addon settings to perform automatic checks for the new versions of Blend4Web.

- API to control Motion Blur postprocessing effect.

New methods to control Motion Blur effect `get_mb_params()` and `set_mb_params()` has been added to `scenes.js` module.

- Support for Timeline markers.

To extract frame numbers from timeline markers a new method `marker_frame()` has been implemented in `scenes.js` module.

- New NLA APIs.

A new set of methods: `set_range()`, `reset_range()`, `set_cyclic()` and `clear_callback()` has been added to `nla.js` module. Also, it's now possible to specify callback in `play()` method.

- New API to change Canvas resolution.

To change Canvas resolution it's sufficient to execute method `set()` from `config.js` module with the following parameters: `canvas_resolution_factor` and `value`, where `value` is the new resolution of Canvas. This feature is particularly useful for creating high-definition screenshots.

- Support for Vertex Groups -> Length option in Hair particle system.
- New API documentation.

To document types used by Blend4Web applications we developed a new formal type system. For example, in our old API docs 3D vectors had `Float32Array` type. Now they have formal `Vec3` type. This solution allows us to formulate more clear and intelligible descriptions for API and, more importantly, helps our users to develop more readable and reliable applications.

- New colors API.

To work with colors in efficient way two new modules: `rgb` and `rgba` have been created. They include APIs to create new color vectors and convert them between different profiles.

- Automatic determination of path to SDK.

Addon option Blend4Web SDK Directory is filled up automatically, if the addon is located in default `blender_scripts` directory inside SDK.

- API for correct calculation of Canvas 2D coordinates.

For proper manipulations with mouse cursor and touchscreen devices the engine requires correct 2D Canvas coordinates.

Details about calculation and use cases of such coordinates are described in the separate [topic](#).

To support this feature the following methods have been added to engine's APIs: `client_to_canvas_coords()`, `set_canvas_offsets()`, `update_canvas_offsets()` and `force_offsets_updating()`. Also, there is a new configuration option: `track_container_position`.

- API to change smooth factors of the camera's movement.

To change smooth factors of the camera's movement the following APIs have been implemented: `set_plock_smooth_factor()` and `get_plock_smooth_factor()` in `mouse.js` module (for Pointer Lock mode) and also `set_camera_smooth_factor()` and `get_camera_smooth_factor()` in `app.js` addon (for general use cases).

- New favicon picture has been added to Webplayer.

Changes

- New API spec for collision detection and ray casting.

Methods `append_collision_test()` and `append_ray_test()`, as well as `create_collision_sensor()` and `create_ray_sensor()` now have new spec, which is incompatible with the previous versions of Blend4Web engine. All developers should consider upgrading their applications to match this new behavior.

- Rendering to texture changes.

It's now possible to render scene into texture cyclically, e.g. when two scenes render one into another. The main constraint here is the requirement to have at least one scene, into which is nothing is being rendered.

- The option Apply Default Animation is now disabled if an object already have an NLA animation attached.

In cases when the object already have an NLA animation attached, the option Apply Default Animation is disabled to eliminate possible animation conflicts.

- Changed Hemi lamp behavior.

If the object is being illuminated by the Hemi lamp, the Lambert shading model will always be applied to its materials. This is done to match the Blender's behavior.

- Support for exported-to-HTML video textures in Firefox browser.

Since Firefox 38 an error with video textures exported to base64 was fixed, so it's now possible to use them in such browsers.

- Changed assignment of UV layers to match Blender's behavior.

Missing from node materials UV layers are determined automatically as it's done in Blender.

- Improved stability of exported to HTML video textures.
- Optimizations of CSM shadows.
- Depth shader optimizations.
- Billboard objects optimizations.
- Configuration option `resolution_factor` was renamed to `render_resolution_factor`.
- Improved support for particle emitters which have several materials.

Distribution of the particles to emitters as well as `vertex colors` inheritance is being done taking into account each material i.e mesh part.

- Changed Wind Bending inheritance on particle systems.

If Wind Bending Inheritance property is set to Instance then Wind bending property for emitter object is not switched off anymore.

- Updated messages about addon/engine version incompatibilities.

For more info see [version errors](#).

- Remove sensor locks API.

Unused sensor locks API was removed from controls.js module.

- Fixed behavior of node materials with missing Output node.

Fixes

- Fixes in screenshooter.js addon.

Fixed an error with impossibility to take a screenshot.

- Fixed a bug in set_frame() method from nla.js module.

Fixed set_frame() inaccuracy.

- Improved exported stability.

- Fixed a bug with addon removal/update on Windows.

Refined binary loader in addon.

- Fixed a bug with shading from SPOT/POINT lamps.

- Fixed incorrect behavior of coordinate calculations in methods get_coords_x() and get_coords_y() from mouse.js addon.

- Fixed calculations of alpha channel in Outline effect.

- Fixed Wind Bending effect error.

- Fixed an error when particle's Scale was not taken into account on particle systems.

- Fixed synchronization error on animated EMITTER particle systems.

- Fixed a bug with shadows on billboard objects.

- Fixed incorrect exporting of Override Mesh Boundings settings.

- Fixed a bug with billboard rendering on iPad.

Known Issues

- Problems with updating of the add-on.

It's strongly advised to restart Blender after installing a newer version of Addon/SDK.

- NVIDIA 331 driver in Linux can cause WebGL errors.

- There is a [bug](#) with video textures on Chrome 43 for Android.

Please update your Chrome browser to Beta or wait until the next Chrome update.

v15.05

New Features

- Glow effect.

Supported [an effect](#) which occurs when the light scatters in the atmosphere and inside of the human eye and looks like a halo around glowing objects.

- The local development server runs automatically.

A new option has been added to the addon settings. This option turns on automatic start of [local development server](#) upon opening Blender. With the help of this functional web applications in development can be run without any preparations.

- Cube reflections.

Apart from plane reflections, there are now cube reflections available. There is a new Reflection Type option available when Reflective flag is set on the object. Setting it to Cube turns this feature on.

- More NLA options supported.

Added support of Blender's NLA tools: Scale, Muted, Reversed and Repeat. The support of these tools broadens capabilities of interactive scene developers.

Furthermore, to control NLA through API a new `nla.js` module was added. This module contains methods like `play()`, `stop()`, `get_frame()`, `set_frame()` that can play/stop NLA and get/set the current frame. New methods have been involved in implementing control panel interface in the Viewer app.

- Increased rendering possibilities for sky textures.

Influence parameters for sky texture rendering are now supported. Those parameters are: Blend, Horizon, Zenith Up, Zenith Down, “Blend”, Negative, RGB to Intensity, DVar.

- In node materials, the engine now correctly processes connections between inputs and outputs of different types.

It is now possible to connect outputs of one type with inputs of another type ([Non-critical error](#)). Now Blender's native behavior is supported by the engine.

- For Hair particles the options on the Rotation panel are now supported.

Now the engine fully supports the state of particles set in Blender. In particular not only location and scale are supported now, but rotation as well.

- Some demos for postprocessing effects demonstration were added.

Examples were prepared for the following effects: Bloom, Depth of Field, God Rays, Motion Blur and SSAO.

- Added a new module container.js.

DOM tree elements can be added with a specific depth relative to the depth of the canvas element with the help of container.js's insert_to_container() method. This method replaces CSS property z-index because location depth of elements is now determined by their position in the container element.

- Improvements in the physical engine.

Margin property of physical elements and materials is now supported. This option allows for improved stability of object collision simulation. Bullet engine was updated to version 2.83.

- API for changing camera controls mode.

Added methods of [changing movement style of the camera](#). The control mode switch example can be found in [Code Snippets](#) app in the “Camera Move Styles” section. Also has been added set_hover_pivot() method. This method allows to shift the control point of HOVER camera.

Changes

- Reorganized SDK's scene list.

All the scenes in the Viewer app has been sorted by groups: App contains finished apps, Demos contains demo-scenes and examples, Tutorial Exports contains tutorials source files.

- Added syntax highlight in Code Snippets app.

Code Snippets app now has a new design. Also it was optimized for low-resolution screens.

- API controls of video and canvas textures have been changed.

A new parameter data_id has been added to these textures' control methods. This parameter contains an ID of dynamically loaded scene.

- Handling of animated bone excess has been changed.

The skeletal animation now just turns off when exceeding the maximum number of bones. It resulted in shader compilation error and unstable application behavior before.

- Some particle system properties have been renamed and now they are turned off by default.

In particular, Hair particle system's properties Randomize Location And Size and Randomize Initial Rotation are turned off by default now.

- Doppler effect for speakers is now turned off in some browsers.

Doppler effect support in Web Audio has been pronounced as deprecated and will be removed in Chrome starting from version 45. Other browsers still support this functionality.

- Changed objects' behavior when both skeletal and vertex animation are applied.

If an object has both armature modifier and vertex animation applied on it, the armature modifier won't be exported.

- Rendering of particle system procedural animation (Wind Bending effect) has been optimized.
- The main .json and .bin scene files loading error handling has been improved.
- Windows 32 version of Blend4Web addon is now compiled natively.

This feature improves compatibility of the add-on with such systems.

Fixes

- The error that led to the wrong height of the description element in module "anchors.js" has been fixed.
- Support for Epiphany and other WebKit-based browsers.

Achieved by fixing the code which works differently in the different JavaScript engines.

- Gestures on Internet Explorer 11 were disabled for Microsoft Windows touch devices.

Previously, gestures usage (Windows Touch Gestures) was leading to unnecessary HTML-elements scaling and movement on such configurations. It is expected that correct gestures' behavior will be supported in further browser releases.

- Vertex animation with animated armature bake error was fixed.
- The error with rendering billboard objects on the iPad has been fixed.
- Node material's NLA animation applied to several objects was fixed.
- Fixed a bug related to the Motion Blur effect.

Known Issues

- Problems with updating of the add-on.

It's strongly advised to restart Blender after installing a newer version of Addon/SDK.

v15.04

New Features

- Deformations by using Shape Keys (Morphing).

Added support for object's `Morph targets` (known as `Shape Keys` in Blender). To apply such keys, use the `set_shape_key_value` method of the `geometry.js` module. Simple example of how to use such functionality is given in the [Code Snippets](#) app.

- Support for Horizon Color and Zenith Color background settings.

It's now possible to tweak scene background by using the Horizon Color and Zenith Color properties as well as the Paper Sky, Blend Sky and Real Sky options directly from Blender.

- Support for the Gamma node.

We have implemented the Gamma node back in Blender v2.74. Now this node is finally supported by Blend4Web.

- Various improvements in the Anchors tool.

It's now possible to limit the pixel size of an annotation. Added support for dynamic loading/unloading of Anchors. Implemented the possibility to hide and show Anchors by using the `show()`/`hide()` API functions and/or by the NLA Script tool.

- Shader optimizations.

Shader compiler improvements. Added the following features: local variables optimizations, brackets removal. Improved the performance of node materials.

- Physics engine optimizations.

To save the load time, the physics modules are now loaded only when explicitly required. Overall size of the modules has been decreased by 20%.

- Extended tools for physics debugging.

The new `physics_stat()` method has been added to the `debug` module. This method returns physics statistics such as the number of physics objects (separated by type), amount of geometry and other info. It is now also possible to display the number of physics iterations per second aka Physics FPS (activated in the config module).

- The new API method to attach objects to the camera independently from the camera's aspect ratio or the field of view.

Implemented in the `append_stiff_viewport()` method of the `constraints` module.

- The new module to perform transformations: "tsr.js".

This new module makes it possible to apply a variety of transformations to objects by using versatile TSR vectors. Each TSR vector combines translation, scale and rotation (hence the name). These vectors may be used instead of matrices as a more convenient and effective way to apply transformations.

- The possibility to exclude any directories from being converted by the resource converter.

To exclude some directory from being converted by the `resource converter`, it is enough to place a file named `.b4w_no_conv` to this directory.

Changes

- The API documentation has changed its appearance. Links for quick access to methods and properties were added.
- Skeletal animation now takes into account the relative translation of an armature and a skinned object.

Native Blender's behavior is now supported. There is now no need to position an armature and an animated object in the same place and with the same rotation and scale.

- Independent translation, rotation and scale animations are now supported.
The animation system no more forces keyframes to be present in every channel, which makes it possible to save original values in unused channels or change them with API.

- World background support.

Background and sky can be enabled with the Sky Settings > Render Sky option under the World tab. It is turned off by default.

- The Uranium physics engine now consists of two modules.

There are now two physics modules instead of one: `uranium.js` - the engine's code and `uranium.js.mem` - the file for memory initialization. Both modules must be placed in the same directory.

- The glow effect together with its related components was renamed to Outline.

The new name better describes the principle of this effect: highlighting objects' edges.

- Limiting the camera translation using the “append_semi_stiff_cam” method of the “constraints.js” module is now possible only for the “EYE” type.
- Local Development Server in Blender changes.

Now, instead of the standard Python SimpleHTTPServer, the Tornado web server is used as the [local development server](#). The new server has greater performance and it also offers more options to disable browser cache.

- Keyboard controls for sliders in the Viewer application were added.

You can now control sliders with < and > keys.

- Changes in the “update_object_animation” method of the “animation.js” module.

The optional “force_update” parameter was added. It forces animated objects to be updated even when their animation is not being played back.

- API changes in the mouse.js module.

The enable_mouse_hover_outline and disable_mouse_hover_outline methods were added.

The following methods were declared deprecated and will be removed in the next releases: enable_mouse_hover_glow and disable_mouse_hover_glow.

- API changes in the scenes.js module.

The outlining_is_enabled, set_outline_intensity, get_outline_intensity, apply_outline_anim, apply_outline_anim_def, clear_outline_anim, set_outline_color and get_outline_color methods were added.

The following methods were declared deprecated and will be removed in the next releases: set_glow_intensity, get_glow_intensity, apply_glow_anim, apply_glow_anim_def, clear_glow_anim, set_glow_color and get_glow_color.

- Some settings were changed in the Object > Blend4Web panel.

The Enable Outline option was added to enable using the [outline effect](#) on the given object. Also, the Outline on Select option was added to activate glow animation when the object is selected (previously this behavior was defined by the Selectable flag).

- New settings were added to the Scene > Blend4Web panel.

The Enable Object Outlining option was added to control the overall possibility of outlining. Similarly, the new Enable Object Selectable option controls the overall possibility of objects’ selection.

- Now the following object properties: Apply Scale, Apply Modifiers, Export Vertex Animation, Export Edited Normals and Export Shape Keys are mutually exclusive.

- API changes in modules.

The new `is_armature` method was added to the API of the `util.js` module. It checks if the given object is of the ARMATURE type.

The new `get_parent` method was added to the API of the `constraints.js` module. It returns the parent object of the given object.

Fixes

- Fixed a bug in the “anchors.js” module which caused objects’ descriptions to disappear.
- Fixed a bug in the Animation Baker script that occurred when there were armature objects in hidden layers.
- Fixed the camera’s behavior while using “`append_semi_stiff_cam`” method of the “`constraints.js`” module.

Fixed correction of the camera’s vertical axis relative to the parent object. Also the original camera orientation is now being taken into account. This can require some adjustments of the camera’s rotation limits that are passed to this function.

- Fixed a bug with reloading of the playlist when it was empty.
- Fixed the buggy behavior of physical objects that occurred after deleting at least one of them from the scene.
- Fixed a bug that occurred when there were zero-scaled objects instanced through DupliGroups.
- Fixed a compilation error of the water shader occurred on Windows and some mobile devices.
- Fixed a bug that occurred when there were duplicates of animation keyframes.
- Actions from different files sharing one name can be now used for NLA animation.
- Fixed duplication of event listeners that occurred when the “pointerlock” function was repeatedly called.
- Fixed behavior of the “Alpha sort” transparency type for dynamic objects.
- Fixed an add-on compilation error that occurred on Windows without C++ 2010 runtime installed.
- Fixed a bug with billboard rendering on iPad.

v15.03

New Features

- New tool for adding annotations to 3D objects.

Now it is possible to assign anchors to empty objects. These anchors can be of three different types: “Annotations” - information from object’s meta-tags is used, “Custom Element” - a custom HTML-element from the current web-page can be used as annotation, “Generic” - an invisible anchor with coordinates calculated using anchors.js module API.

- Animation and API methods for Value and RGB nodes in node materials.

Now it is possible to animate not only Value nodes but also RGB nodes. Also, the corresponding API methods for changing such nodes were added in the objects.js module.

- New “Code Snippets” application.

This [application](#) was created to simplify access to the examples of engine’s functionality. It is also possible to look at the examples’ scripts. This application can be launched from the index.html file located in the Blend4Web SDK’s root directory.

- New control functions for the Glow effect.

New APIs were added in the scenes module: `get_glow_intensity()` and `get_glow_color()`.

- Improvements in the Scene Viewer.

Design of the “Home” button was changed. A new button “All objects selectable” was added. It allows to turn off automatic “Selectable” option assignment for all scene objects. Also, it is now possible to see the total number of shaders on the loaded scene.

- Dynamic copying of scene objects (instancing).

Now it is possible to dynamically [copy](#) and [remove](#) scene objects (to create and remove instances).

- Handling errors related to the B4W_PARALLAX Blend4Web-specific node.

In case of incorrect usage of the B4W_PARALLAX node, an [export](#) error warning is generated.

- New options in the application’s builder.

There are now new options in the application builder: `-j` and `-c`. They add scripts and styles correspondingly to the exceptions in order to be not compiled.

- Experimental Blend4Web render engine.

It can be turned on in the addon settings using the “Register Blend4Web render engine (Experimental)” flag. This mode is designed to simplify customization of scene properties. Also, it simplifies the interface by removing unsupported panels. At the moment, it is not possible to edit shader node tree in the Blend4Web render mode.

Changes

- Origin for counting off the camera limits has been changed.
Setting camera movement limits via API now perfectly corresponds to values measured in the engine’s coordinate system. Setting horizontal limits for the TARGET camera in the Blender’s world space has been changed. Thus, it may require changeovers for old scenes.
- Documentation update for the camera settings.
- Horizontal and vertical limits of the camera rotation are completely independent from each other.
- Some APIs in the camera.js module were changed.

APIs of the camera.js module have undergone a number of changes.

New methods were added: `is_target_camera`, `is_eye_camera`, `is_hover_camera`, `rotate_camera`, `rotate_target_camera`, `rotate_eye_camera`, `rotate_hover_camera`, `get_camera_angles`, `hover_cam_set_translation`.

The following methods were declared as deprecated and will be removed in the next releases: `rotate_pivot`, `rotate_hover_cam`, `rotate`, `set_eye_params`, `get_angles`, `translate_hover_cam_v`, `set_hover_cam_angle`.

The `set_ortho_scale` and `get_ortho_scale` methods now print error message when they are applied to the Orthographic camera. The behavior of the `get_hover_angle_limits` method was also changed. This method now returns angle limits for the HOVER camera in the [down, up] format instead of [up, down] as it was before.

- The lighting system was significantly optimized.
Many of the lamp props are calculated at the compile time now. Now, there is no 4 lamp restriction is imposed for some mobile devices.
- The HTML layout method was changed for apps using the app module.

Now, upon initializing an application using the `app.js` module, the dimensions of the created `<canvas>` element are completely determined by the size of the container element. Thus, if a `<div>` element is used as a container, the size of `<canvas>` will be zero by default since `div`’s default size is zero. You can set correct parameters for the container with CSS and inline-style. Also, you have to use `resize_to_container()` method

from the app module when the container is changed. The same effect may be achieved if the autoresize option is set upon initializing the application (in the app.init() function). The low level method for changing the element's dimensions with main.resize() function is still supported.

- Now, the enable_controls() function from the app module should be called without any parameters.
- Deprecated API methods were removed.

`textures.js` module : `stop_video`.

`scenes.js` module: `add_object`, `get_screen_scenes`, `set_light_pos`, `set_light_direction`, `set_dir_light_color`, `get_lights_names`, `remove_all`, `check_collision`, `check_ray_hit`, `get_appended_objs`, `get_object_by_empty_name`.

`physics.js` module: `set_character_dist_to_water`.

`material.js` module: `set_batch_param`, `set_max_bones`, `max_bones`.

`main.js` module: `set_shaders_dir`, `set_texture_quality`.

`data.js` module: `get_bpy_world`.

`controls.js` module: `sensor_make_positive`, `sensor_make_negative`.

`camera.js` module: `change_eye_target_dist` (deprecated `MS_CONTROLS` was also removed).

- There are now new conditions to allow changing object's position via API.
The functions from the `transform.js` module related to changing object position can be now applied to `dynamic objects` only.
- You can now use TEXTURE nodes without textures.
In this case the rendering of the node material completely corresponds to Blender.
- Updated the procedure of compatibility checks for versions of exported files and the engine itself.
The engine will report about the scene's incompatibility by printing `messages in the browser console`.
- The “Do Not Batch” property was renamed to “Force Dynamic Object”
This option instructs the engine that the object must be dynamic regardless of other settings. Now its name is more clear.

Fixes

- Fixed camera autorotate feature of the web player.
- Fixed an error related to the fullscreen mode in the Web Player.

- Fixed an error related to the determination of the camera's angular coordinates in some positions.
- Fixed an error with camera autorotation if the horizontal limits are enabled.
- Fixed an error when Blend4Web-specific nodes were being added multiple times to a .blend file.
- Fixed a bug with replacing materials by using the “inherit_material” function from the “material.js” module.
- Fixed an error occurred while rendering reflections on an object which has been changed through the “material.js” module APIs.
- Fixed generation of the debugging wireframe spheres.
- Fixed optimization of the TEXTURE nodes in node materials.
- Fixed “Clamp” option behavior in the MixRGB (Linear Light) node.
- Fixed an export error occurred when an object shares its mesh with another object and one of the following flags is set to true: “Apply Scale”, “Apply Modifiers”, “Export Vertex Animation” or “Export Edited Normals”.
- Fixed an error with “Blend4Web > Preserve Global Orientation and Scale” option on some mobile devices.
- Fixed fog rendering error in some versions of Chrome/Firefox under Windows.

Known Issues

- Problems with updating of the add-on.
It's strongly advised to restart Blender after installing a newer version of Addon/SDK.
- Video textures do not work in Firefox for scenes exported as HTML files.

v15.02

New Features

- The local development server can be run from Blender.
It's possible now to run local development server when using Blend4Web SDK. This server allows fast access to the Blend4Web SDK content and also make it possible to [automatically open exported scenes](#) in the Viewer application.
- Support for Spot lights shadows.

Shadows for Spot lamps were processed in the same way as for Sun lamps. Now the calculations are performed in the same way as it's done in blender i.e the light scattering is taken into account.

- Added/improved support for “Metaball”, “Surface” and “Curve” objects.

Support for Metaball, Surface and Curve objects was added. Objects of these types are automatically converted into meshes during export. Support for Curve objects in modifiers was preserved.

- Social networks buttons are added to the Web Player.

These buttons allow placing a link and a description to the 3D scene in one of the four popular social networks.

- Added support for editing the list of Viewer scenes directly in Blender.

It is now possible to [edit assets.json file](#) with a list of Viewer scenes inside Blender. This works only when using Blend4Web SDK.

- Screenshots can now be taken in the Viewer application.

- New `fallback_video` option is added to the Web Player.

Now the user can choose a video file to play instead of 3D content on systems without WebGL support. It is possible with the help of the new `fallback_video=/path/to/video/` option.

- Improved rendering to texture functionality.

Added support for rendering scenes into several textures at a time. Scenes now can have any nesting level.

- Billboards now can save orientation and scale in world coordinates.

To use the feature you need to set [Blend4Web > Preserve global orientation and scale](#) flag in the object's settings panel.

- Improvements on the main SDK web page.

It is now possible to find out the version of the SDK and check the system for WebGL compatibility on the main web page.

- Added support for the Clamp flag in MATH and MIX_RGB nodes

At first this functionality was released in Blender 2.73, and now it's also supported in the engine.

- Considerable improvements in rendering quality on systems without depth-texture support.

Supported rendering features on systems without depth-texture support were extended. There are such effects available now: reflections, bloom, glow, motion blur, anti-aliasing.

- New documentation is added.

Added documentation for the vec3, vec4, quat, mat3, mat4 modules and for the global namespace b4w. Documentation web pages design was improved.

- Support for several engine instances on the same web page.

Several engine instances can now work simultaneously, by specifying the namespace on engine's initialization stage.

- Possibility to use SDK on Apple OS X.

On OS X all SDK functionality including engine and applications building, resource conversation and documentation generation is now available.

- The new `set_trans_pivot()` method is added to the camera module.

This function allows setting an arbitrary position of the pivot point and the position of the camera of the Target type.

- A new “version” property is added to the “project.py” utility.

This property allows adding a version to the scripts and styles of the compiled application.

Changes

- Now it is possible to add options with the same names via browser address bar.

A new optional parameter `allow_param_array` is added to `get_url_params()` function of app module. It is set to false by default. Setting this parameter to true leads to a merge of identical functions into a massive, other way the last one will be used.

- Improved “Background Music” speaker.

Now the user can specify parameters of delay and playback time.

- Blend-file now includes special Blend4Web nodes by default.

Now there's no need to add [Blend4Web special nodes](#) into a file. It is available in both SDK and addon versions of Blend4Web.

- Changed export of empty “Mesh” objects.

Now Mesh objects without polygons are exported as Empty.

- Changes in the “light.js” module.

Added function `get_light_type`; functions `get_light_params` and `set_light_params` now get object LAMP instead of objects name. Also the user now can change `spot_blend`, `spot_size` and `distance` properties of the SPOT light source through those functions.

- Improved refractions on LOW quality settings.

New simplified (without distortion) refraction model is now used when LOW quality is chosen.

- Shader nodes optimization.
- Now automatic camera rotation can be turned off by touching touch screen.

Fixes

- Corrected behavior of the preloader for the Web Player.
Some artifacts could be visible on B4W logo while opening Web Player.
- Fixed an error in rendering of one-cascaded shadows. The error was related to appearing of a hard non-lighted line on the cascade border.
- Fixed an error related to names collision while linking objects in Blender.
- Fixed an error with optimization of SDK apps.
- Fixed export error of flat shaded meshes on Linux x32.
- Fixed incorrect behavior of Target camera in particular cases.
- Fixed an error when using a shadow map with size exceeding device limits.
- Fixed an error that leads to FPS drop in Firefox 35/36 on Windows when shadows are turned ON.

Known Issues

- Problems with updating of the add-on.
It's strongly advised to restart Blender after installing a newer version of Addon/SDK.
- Video textures do not work in Firefox for scenes exported as HTML files.

v15.01

New Features

- Support for panning on touchscreen devices.
Panning is performed by swiping two fingers on the surface of the screen.
- Support for “Text” objects.
These objects are now automatically converted to meshes during export.
- Extended support for the NLA Script tool.
Added new logic slots: Show Object and Hide Object used for hiding and showing the objects, Page Redirect - for redirecting to other webpages , Page Param - for storing any webpage parameter in given numerical

register. Simplified usage of Select & Jump and Select & Play slots. Now it's not required to specify Selectable property on selectable objects.

- Support for high definition displays (HiDPI, Retina).

The HiDPI mode allows to achieve considerable improvement of picture quality on devices with high resolution. This mode is activated automatically upon application startup if ULTRA quality profile has been selected. If necessary, high resolution can be turned on for other quality profiles.

- Support orthographic camera scaling.

An API to change the Orthographic camera scale has been added (Orthographic scale in Blender).

- “autorotate” option has been added to the webplayer.

The [option](#) autorotate is used to turn on the automatic camera rotation as soon as the scene loads.

- Simplified keyboard control mode has been added to function “enable_camera_controls”.

The mode is enabled by passing the optional parameter disable_letter_controls. Thus, the keyboard controls with letter keys (WASD and so on) will be turned off. This feature can be used in cases when you need to use the letter keys for purposes other than moving the camera.

- Support for gyroscope on mobile devices.

To work with gyroscope on mobile devices the two sensors were implemented. The first sensor allows to operate with current device position compared with the previous one (position delta). It's created by using the function `create_gyro_delta_sensor` from “controls.js” module. The second sensor returns current device angle and created by `create_gyro_angles_sensor` from “controls.js” module. It's worth to mention that all angles are given in radians. Also the special addon “gyroscope.js” was created. This addon implements simple camera movements due to device rotation. You can find an example of using such feature in our Viewer application by selecting the “Gyroscope” menu option.

- New “Do not Render” property has been added to material settings.

Enabling the property allows to hide parts of the scene objects which use such material.

- Support for video-textures on IE 11 and iPhone.

The support is achieved by creating a new video-sequence format, *.seq. For more info check the following [topic in documentation](#).

- Support for “title” tag in Web Player.

The Web Player's title (shown as web browser header) is now extracted from JSON file of the loaded scene. For more info about this feature

check the following topic in documentation.

- Support for meta tags in Blender.

It's now possible to append meta tag information to scenes and objects in Blender. Possible tags for scenes are "title" and "description". Possible tags for objects are "title", "description" and "category".

- Added support for execution of user-defined functions every frame.

To help users to create sophisticated application the new function `append_loop_cb` has been added to "main.js" module. This function allows to execute given callback every frame. This callback has two parameters: time since the application start and time delta between current and previous frame. Both parameters are in seconds. To cancel the callback execution every frame you should remove it by using the `remove_loop_cb` function from the module "main.js".

- Added support for simple preloader screen animation.

To create an application with animated preloader pass an option "preloader_fadeout" with the value "true" to the function `create_simple_preloader` from "preloader.js" module.

- Added support to export converted media files to HTML file.

Now then you export HTML files it's possible to store converted files in them. To do so you need to enable "Export Converted Media" option in the [export options](#).

- Added support for using min50 and dds textures in Web Player.

To enable this feature pass "compressed_textures" parameter to Web Player.

Changes

- SDK file hierarchy was simplified.

The external directory was removed, all its content was moved to the upper level - in the root directory of the SDK. The file with the list of the scenes used by the Viewer application is now located in `apps_dev/viewer` directory.

- Changed camera autorotation behavior (experimental).

If the camera limits are present, the camera smoothly slows down when approaching the limits, then moves in the opposite direction.

- The usage of "Special: Collision" property was changed.

Earlier, enabling the option automatically resulted in objects' hiding. Now, to do the same thing, you have to specify Do not Render property in the material settings.

- Changed suffix for converted media files.

Old *.lossconv.* suffix was replaced by *.altconv.*.

- Behavior of “Do not render” object property was changed.

Now, when the option is activated, an object’s physics is not disabled.

The object simply becomes invisible.

- Improved the SDK structure.

Free and Pro SDK now come with the new and more polished examples; old and arid examples were dropped.

Fixes

- When using “Panoramic” camera type in Blender the camera automatically obtains “Perspective” type when exported.
- Fixed bug with “Target” camera dragging in rare cases.
- Minor fixes in the “B4W Anim Baker” addon.
- Fixed issue with sound for scenes with multiple cameras.
- Improved stability of “Timer” sensors in “controls” module.
- Fixed issue when browsing exported HTMLs in IE browser.
- Video texture optimizations. Now the video texture is not updated for suspended video playback.
- Fixed rendering issue in node materials with “REFRACTION“ node.

Known Issues

- Problems with updating of the add-on.

It’s strongly advised to restart Blender after installing a newer version of Addon/SDK.

- Video textures do not work in Firefox for scenes exported as HTML files.
- Slow and unstable rendering of depth textures in Firefox 35.

In various scenes the FPS is degraded when shadows are turned on. There is also an incorrect rendering of transparent materials. The issue is reported [here](#) and is to be fixed in the future browser updates.

v14.12

New Features

- Camera velocity settings are now available.

Now `camera movement velocity` can be set up, including translation, rotation and zooming. Velocity for all camera types (Target, Hover, Eye) can be tweaked both using Blender UI and through Blend4Web API.

- Mipmapping is now supported for Canvas textures.

We have added mipmapping support for Canvas textures.

- Full support for the “MAPPING” node.

Now all Vector type options available for the MAPPING node are supported, including Texture, Point, Vector and Normal.

- Glow on mouse over.

API in the `mouse.js` module were created for the effect of outlining the objects under the mouse pointer. Also, for purposes of controlling this effect, `enable_mouse_hover_glow()` and `disable_mouse_hover_glow()` methods were added. The objects should have Object > Blend4Web > Selectable checkbox enabled.

- A brand new app building system.

Now, the users can develop their apps right in the SDK thanks to the new `project.py` utility. [This script](#) makes it possible to build the apps together with the engine, to minify JavaScript and CSS files and to export the final apps to be deployed on a server.

Changes

- Removed support for deprecated “UV translation velocity” texture settings.

We recommend to use node materials instead.

- Removed deprecated “Levels of Detail” user interface.

This functionality can be used through Blender’s standard “Levels of Detail” tool.

- The `pointerlock.js` add-on was renamed into `mouse.js`.

- A `mouseup` event is now fired when the mouse pointer is leaving the app viewport.

So the problem with broken camera controls is no longer observed.

- Error message about “Clear parent inverse” is no more shown.

Before, when using parenting, it was required to reset translation, rotation and scale of child objects (Object > Parent > Clear Parent Inverse). Now such transformation is natively supported by the engine.

- “Apply scale” option no longer applies modifiers.

As before, [Apply modifiers](#) should be used in order to apply modifiers.

- Use of normal maps in node materials no longer requires a Material or Extended Material node.

In some cases (e.g. refraction) normal maps can be used in shadeless materials.

Fixes

- Fixed audio playback error occurred when using NLA.

This happened due to insufficient float number precision.

- Fixed incorrect rendering of light sources on mobile devices.

- Layering shadows is fixed when multiple active light sources are present.

Now the shadows are calculated like in Blender, that is areas illuminated by other light sources are not darkened.

- Node material rendering error has been fixed.

The error occurred when a MATERIAL node (or MATERIAL_EXT) with a linked (from another .blend file) material was used.

- Animation baker (“B4W Animation Bake” operator) no longer resets an armature pose.

Now, when using the [animation baking tool](#), the armature pose is being left intact.

- Fixed jerky camera movement upon application startup.

- Fixed error with incorrect determination of the camera’s horizontal movement limits.

- Fixed error occurred when unused textures were exported.

Known Issues

- Problems with updating of the add-on.

It’s strongly advised to restart Blender after installing a newer version of Addon/SDK.

- Video textures do not work in Firefox for scenes exported as HTML files.

v14.11

New Features

- Video textures support.

Video textures are now supported for Image or Movie textures.

- Frame rate.

Frame rate for animation and video textures can now be changed through the Scene > Dimensions > Frame rate option.

- Canvas textures support.

A canvas HTML element can be now used as a [texture](#). The workflow is described in the [manual](#).

- Camera panning.

In the mode when the camera is rotating around a single point (Target) the users now have the ability to move the pivot point within the view plane (so called camera panning) while the right or middle mouse buttons are pressed. This function is turned on by default and can be turned off in Blender settings at need.

- New camera control mode - Hover.

The Hover mode is now available when the camera is gliding over the horizontal plane (including zooming in and out). This camera mode makes it possible to realize scenarios for a convenient viewing of scenes which are spread in two dimensions (rooms, game levels).

- The SDK now contains a root index HTML webpage for simplifying navigation within the distribution.
- The resource converter now has the ability to convert videos.
- We have added a build system which was absent in previous public SDK distributions.
- The export Strict mode is implemented in the add-on.

Activating Strict mode gives the possibility to display all possible errors and warnings connected with incorrect scene settings. The option is useful for final scene debugging for getting the most correct and optimized resource files.

- Audio playback support for iOS devices.

Changes

- The webplayer's "bg" parameter is renamed to "fallback_image".

This option also has changed its behavior. If the fallback_image is defined the error message that WebGL is unavailable is not shown anymore, instead the user sees just the image.

- If there are no sound sources in the scene the sound mute button is no longer shown in the webplayer.
- Generic materials workflow is now more predictable.
- The “mouse_down” sensor provides the code of the mouse button pressed. This code can be obtained from the payload sensor’s parameter.
- Hair particle systems can be now exported significantly faster.

Fixes

- Normal maps now work with Generated and Normal texture coordinates.
Using UV layers is no more required for normal maps.
- Fixed the problem with the wrong path to the physics engine in the webplayer.
This error arose when uranium.js was moved out of the directory containing the main HTML file of the webplayer.
- In the add-on we have fixed the problem with packed textures. Export crashed when the “Automatically Pack Into .blend” option was enabled.

Known Issues

- Problems with updating of the add-on.
It’s strongly advised to restart Blender after installing a newer version of Addon/SDK.

v14.10

New Features

- A new Web Player.

The new minimalistic Web Player design blends perfectly with any 3D scenes. It has a simplified user interface and build-in help. The Web Player works on all devices including mobile ones.

- Improved shadows.

It’s now possible to choose a non-cascaded shadow model, based on a single optimized shadow map. Such model is easier to configure and suits well for relatively small scenes. For more info see the [docs](#).

- Many NLA system improvements.

It's now possible to create a complex logic using the Conditional Jump, Register Store, Math Operation NLA Script logic blocks and register-stored variables.

It's now possible to use all types of supported animations in the NLA, including sound playback, vertex animation and particle emission. It's now possible to play different animation types simultaneously.

- Supported objects billboarding.

The objects received the new set of [options](#), allowing to configure billboarding.

- The “XYZ Euler” mode is supported for animating rotations.

Object and skeletal animations now support the XYZ Euler mode for rotations.

- Support for the GENERATED texture coordinates.
- Support for Cross-origin resource sharing (CORS).
- Scene export process simplified.

The range of material export errors are now not blocking the export. Instead, this material will be highlighted pink at scene loading. Detailed error descriptions can be found in the [manual](#).

- Added support for the “Do not export” option for particle systems.
- Improved stability on iOS devices.

Changes

- Changed SDK path setting for HTML export.

It's now needed to set the SDK path setting for HTML export to SDK root directory. Earlier it was required to provide the full path to embed application. Pay attention, the old behavior is not supported anymore.

- Deprecated the “UV translation velocity” option.

The option will be removed since version 14.12.

- Removed option “Do not export” from the “Object data” panel.
- Removed “Blend4Web > Animation > Cyclic” option from the object properties panel.

Instead, one should use Blend4Web > Animation > Behavior option located in the same place. Scenes with default animations may work incorrectly, so they need behavior property to be set to Cyclic.

- Modified SSAO algorithm realization.

The new implementation is much faster and shows better quality. The settings of the algorithm are changed too. For more info see the [manual section](#).

Fixes

- Fixed rendering error for HALO materials.
- Fixed a rendering error when an object with the enabled “Force Dynamic Object” property has a parent object.
- Fixed error with keyboard shortcuts in Blender.

Fixed error with inability to assign keyboard shortcuts in Blender for export menu items:File->Export->Blend4Web(.json) and File->Export->Blend4Web(.html).

- Fixed crash when loading textures with size exceeding device limits.
- Fixed node material errors resulting in unstable engine behavior.
- Fixed error in node materials that contained complex Node Groups.
- Fixed errors of shaders compilation on devices with mobile graphics Qualcomm Adreno 305.
- Fixed rendering error when using REFRACTION nodes in transparent materials.
- Fixed an issue in “B4W Vertex Anim Baker” tool when current frame reset was occurred after using bake.

Known Issues

- Problems with updating of the add-on.
It's strongly advised to restart Blender after installing a newer version of Addon/SDK.
- Normal maps don't work for Generated texture type.
It is necessary to use UV mapping for normal maps.

v14.09

New Features

- ABSOLUTE type support for the MATH node.
- Support for LEVELS_OF_QUALITY special node.

Allows to control the material's complexity depending on the quality profile which is specified by the user upon engine start.

- Support for SMOOTHSTEP special node.

Simplifies the creation of some effects in node materials.

- Node groups support.

Node groups allow the sharing of node blocks between materials.

- The ability to output intermediate rendering results for debugging.

The rendering result for a certain stage can be now output above the main picture. This can be set up in the config.js module through the debug_subs options.

- The logic for controlling Blender's NLA animation using a visual editor has been implemented.

The NLA Script tool has been added to Blender's interface to allow the implementing of simple scenarios using visual blocks, for example playing an animation in response to the user actions.

- Multiple sensor system improvements.

It is now permitted to register sensor manifolds globally using a controls module method with no connection to any object. To do this null should be passed into the corresponding API. The sensor logic is processed in a more predictable and robust way according to the sequence in which their manifolds are created. Callbacks of the user action events are now assigned using the register_<inputtype>_events() functions. To these functions it is now possible to pass the prevent_default flag which allows to unblock the standard browser behavior for the corresponding events.

- The Web Player now supports physics.

Works only in the Web Player version where JSON files are loaded separately. Physics still not supported in the single HTML files.

- Skeletal animation mixing is now supported.

The animation.js module now contains API for smooth transitions between skeletal animations: get_skel_mix_factor() - for getting the current mixing factor value and set_skel_mix_factor() - for setting it.

- The Value node can now be animated in node materials.

The functionality is similar to other animation types. Working in NLA is also supported.

- Diffuse and Specular lamp's properties are now supported.

- The possibility to render a transparent object above other objects on the scene.

Activated with the Render above all checkbox for transparent materials (i.e. not Opaque).

- Scale is applied automatically to the object mesh.
Activated by enabling the Apply scale checkbox in the object settings.
- High quality profile (including shadows, dynamic reflections and anti-aliasing) has been implemented for iOS.

Changes

- Shadow rendering improved.

Shadow rendering system is significantly changed: it is now based on the Stable Cascaded Shadow Maps technique. This technique allows to greatly diminish the flickering of shadow edges when the camera moves. Smoothing is implemented between cascades. Also shadows of the last cascade fade out at distancing. Softened shadows are rendered using the Percentage Closer Shadows technique. The shadows' user settings are reworked and simplified. Now its possible to tweak the size of shadow maps, blur ratio and the setting for removing self-shadowing artifacts. The new settings are [documented](#) in detail.

- In the Web Player graphics quality settings are now saved independently for each scene.
- The behavior of the app configuration parameters has been changed: physics_uraniun_path, smaa_search_texture_path and smaa_area_texture_path.

These parameters are now calculated automatically depending on the running HTML files location, if they haven't been overridden during the app's initialization.

- Transition is completed to the system of modules which are linked via b4w.require() call.

This also means that starting form the current version its impossible to call modules in the engine's release version using the old b4w.<module> namespaces. For compatibility purposes the ns_compat.js add-on has been implemented, the linking of which allows to restore the old behavior.

- The Web Player's control panel can now be hidden.
- Skeletal animation is now applied to armature objects only.

There is no need to apply skeletal animation to MESH objects. If they are linked to some animated armature, their skinning will be automatic.

- Demos and tutorials are updated according to the newly implemented features.

Fixes

- The preloader didn't disappear in case of a loading error (texture or sound file).

- Lagging during scaling and turning on mobile devices is fixed.
- TARGET-type camera shimmering has been removed for small turnings.
- EYE-type camera controls was fixed for mobile devices.
- The Farm demo controls are improved for Safari browser.
- Errors concerning using the unsupported shading models in node materials are now fixed.
- “Selectable” option now works for the objects without materials.
- There is no longer need to enable “Force Dynamic Object” for the objects that are animated using NLA.
- The particle system error when the object being instanced is parented to another object, has been fixed.

Known Issues

- Problems with updating of the add-on.
It's strongly advised to restart Blender after installing a newer version of Addon/SDK.
- Armature animation mixing doesn't work with some browsers.
If skeletal animation mixing API brings unexpected errors, it is necessary to override standard Math.sign function as follows:

```
var m_util = require("util");
Math.sign = m_util.sign;
```

- genindex
- search

Symbols

3D Modeling, 3

A

add-on, 39

addon

 initialization errors, 50

anaglyph, 219

animation, 263

anti-aliasing, 218

B

background, 207

Blend4Web, 1

Blender, 3

browser, 4

browser technologies, 4

C

Color Correction, 294

compiling shaders, 339

 errors, 343

 import/export directives, 341

 obfuscation, 340

 optimization, 340

 recommendations and limitations, 341

 validation, 340

 WebGL Extensions, 343

crepuscular rays, 214

D

depth of field, 211

diffuse map, 226

DOF, 211

drivers, 5

E

engine, 1

engine features, 5

export, 16, 19

 errors, 52

 warnings about export errors, 56

G

get object, 93

git, 345

 adding and removing files, 349

 checking the status, 347

 commit, 350

 individual settings, 347

 preparing for commit, 348

 resolving conflicts, 353

 synchronization between repositories, 351

 tags, 356

god rays, 214

graphics engine, 1

I

interactive graphics, 5

L

light sources, 196

lighting, 194

M

matcap, 226

material capture, 226

materials, 135

 halo, 144

 lighting parameters, 136

 nodes, 145, 147, 148

 reflection, 139

 rendering properties, 142

 specific parameters, 144

transparency, 138
viewport properties, 143
motion blur, 210

N

normals, 103
editor, 103

O

Object Transform API, 91
outline glow, 216

P

parallax mapping, 227
particle system
 fluids, 241
 instancing, 251

Q

quaternion, 96

R

reflection, 139
 dynamic, 140
 fresnel effect, 141
 static, 140
release notes, 367
render-to-texture, 238
RTT, 238

S

scene, 62
screen-space ambient occlusion, 212
sensor, 326
 manifold, 326
shading, 195
 types, 195
shoreline, 283
shoreline data, 283
skydome, 236
SSAO, 212

T

textures, 222
 diffuse, 226
 environment map, 233
 height map, 227
 mirror map, 235
 normal map, 227
 render to, 238
 settings, 225
 sky, 236
 specular map, 227
 stencil map, 229
 types, 224
three-dimensional engine, 1
transparancy
 settings, 139
 types, 138
transparency, 138

V

version
 errors, 51
video card, 5
viewer, 29
 adding scenes, 19
 launch, 43

W

web player, 33
WebGL, 2
 advantages, 3
 browser support, 2