Godot Editor running in a Web Browser

Rationale

Godot Engine supports exporting games to the HTML5 platform (i.e. browsers). Given that the editor itself is written using the Godot Engine API it should be possible to run the editor in browsers as well.

The reason why the editor wasn’t able to run in browsers up until now was due to some historical web browsers limitations, mostly lack of support for threading, but also file system access. With the introduction of WebAssembly, WebAssembly threads, Javascript SharedArrayBuffer, and an upcoming FileSystem Web API, it is should now be possible to have an almost-native user experience when running the editor on the web.

This will be beneficial in many ways to the engine itself for multiple reasons:

1. It will lower the barrier for new users, which will be able to try out the engine without the need to download anything.

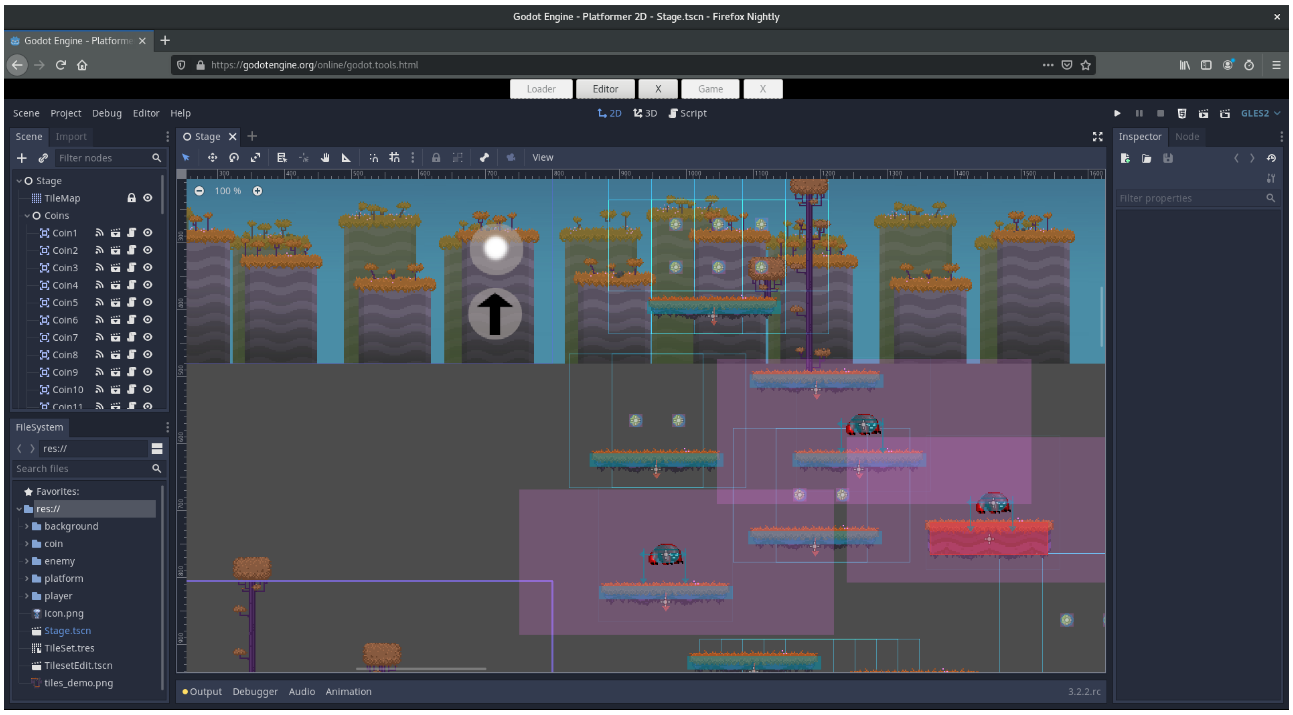
2. Any modification towards reaching that goal will also improve the HTML5 export itself (given that the editor is made like a Godot game).

3. It will allow to use Godot in a reasonable way in environments where installing/downloading applications is not an option (e.g. schools’ computers and tablets), fostering the usage of the engine for educational purpose (which is something we, as an open source community, deeply believe in).

This DOES NOT mean that Godot will move completely to the Web, nor that the Web Browsers version will be the recommended way for professional development, but only as an additional option for cases where it might be useful (again, pick the education sector as an example).

To stress this out: The good-old native editor will always be our main focus.

After this necessary preface, let’s get to the news.



A Godot Editor prototype running in browsers is being presented at https://godotengine.org/online/godot.tools.html

This is a very early stage version, but it allows to run the editor (including the project manager), and make simple projects, while storing the files in either your browser local storage, or an external cloud service (Dropbox is currently supported, but not recommended due to speed limitations. In the future, standard WebDAV support will allow for more providers and better speeds.)

Usage

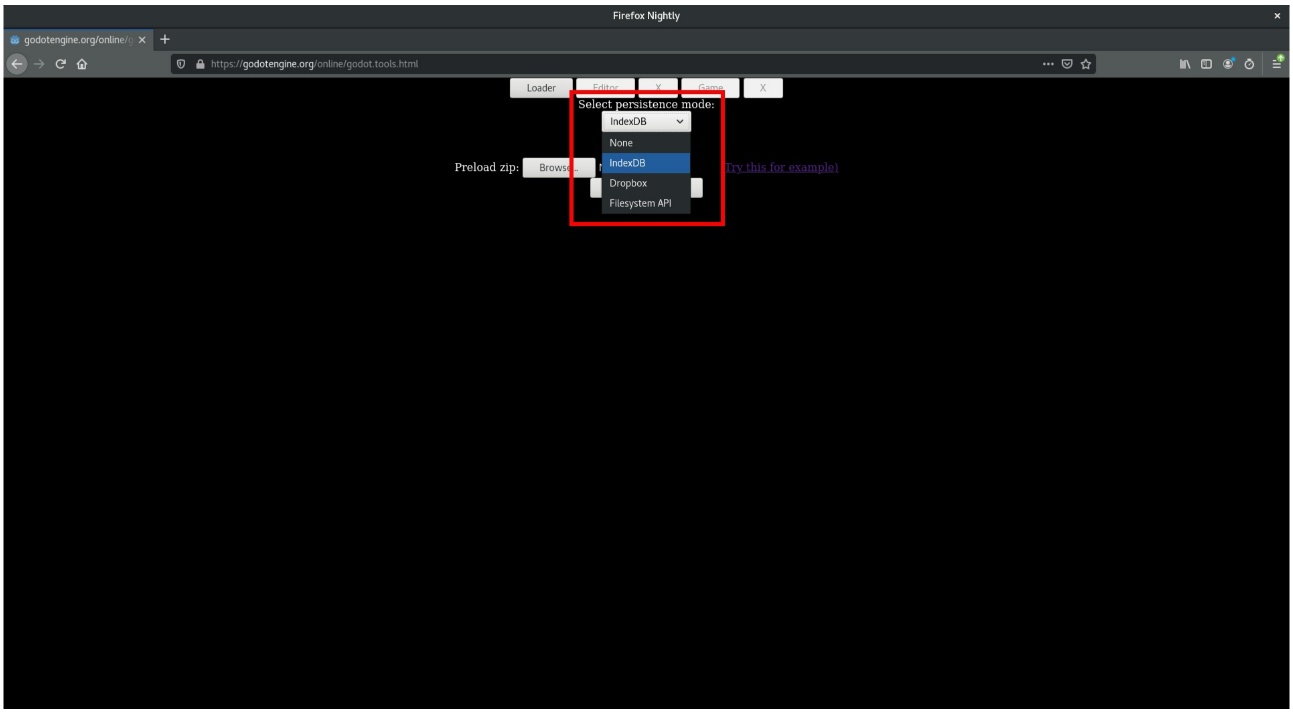
When opening the URL you will be asked how to load the engine, specifically, selecting the persistence support. There are 4 options available:

- None – no persistence will be used, you will lose everything when you close the browser or refresh the page.

- IndexedDB – will use the IndexedDB API to store your files. This is usually limited to 50 megabytes on desktop, and 5 megabytes on mobile (this is the RECOMMENDED storage type for now).

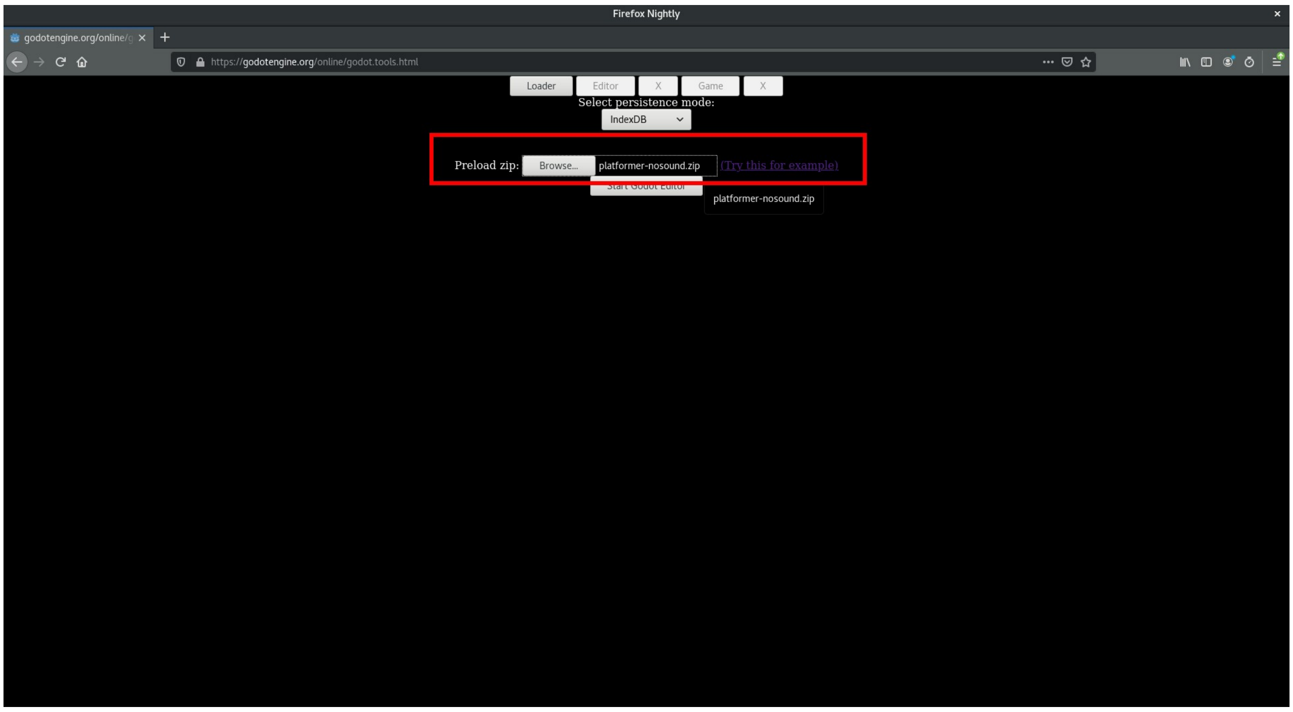
- Dropbox – will store your files in a Dropbox folder, created specifically for the test application. You will also be able to upload files directly to dropbox and they will be available in the engine (after refreshing the page!). This is a very powerful tool, but is currently quite slow, both when loading and saving (it will need to download the whole folder on startup, and changes will be stored asynchronously). Improvements in this area are much needed, and Dropbox support is provided as a proof-of-concept for now and in no way recommended. Cloud support (via standard WebDAV) will in time become the preferred way to use the web editor.

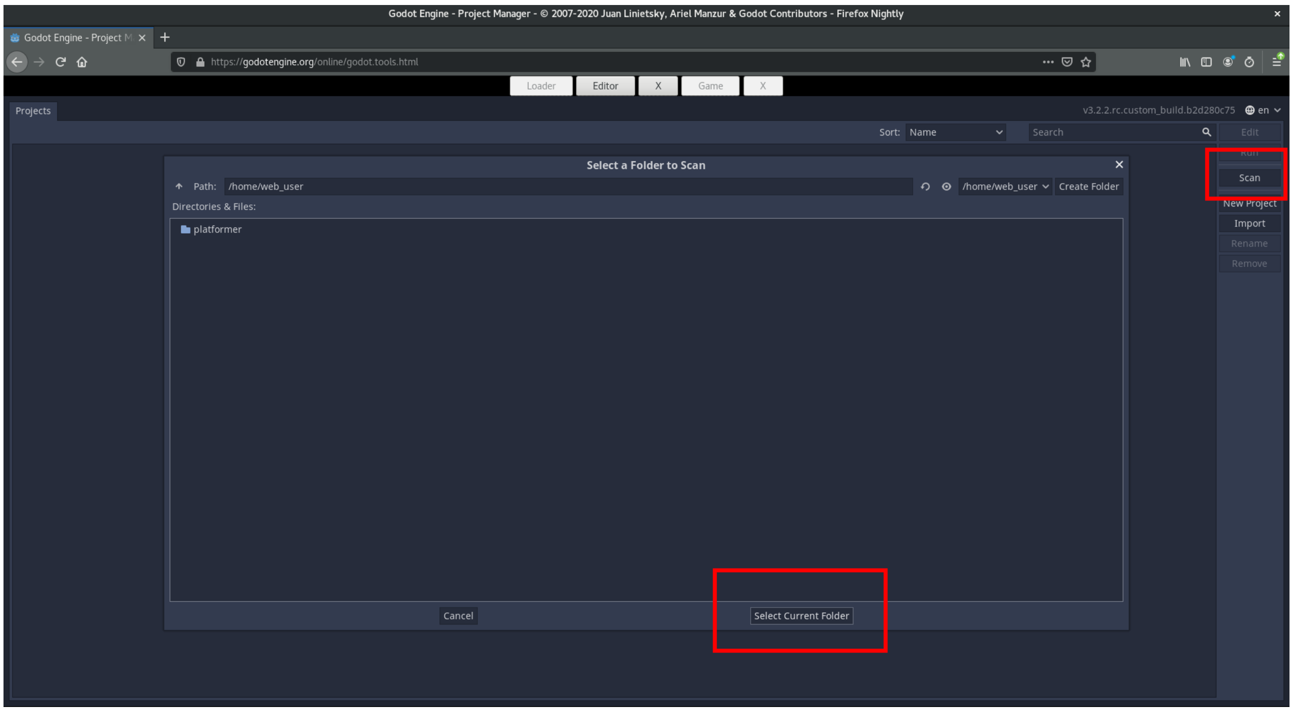
- FileSystem API – will use the new Web FileSystem API, which could potentially expose native file system support in the future, if browser vendors agree on a safe way to do that. This is, again, provided as a proof-of-concept and browser support is very limited for now.



Select persistence mode:
IndexDB
None
IndexDB
Dropbox
Filesystem API

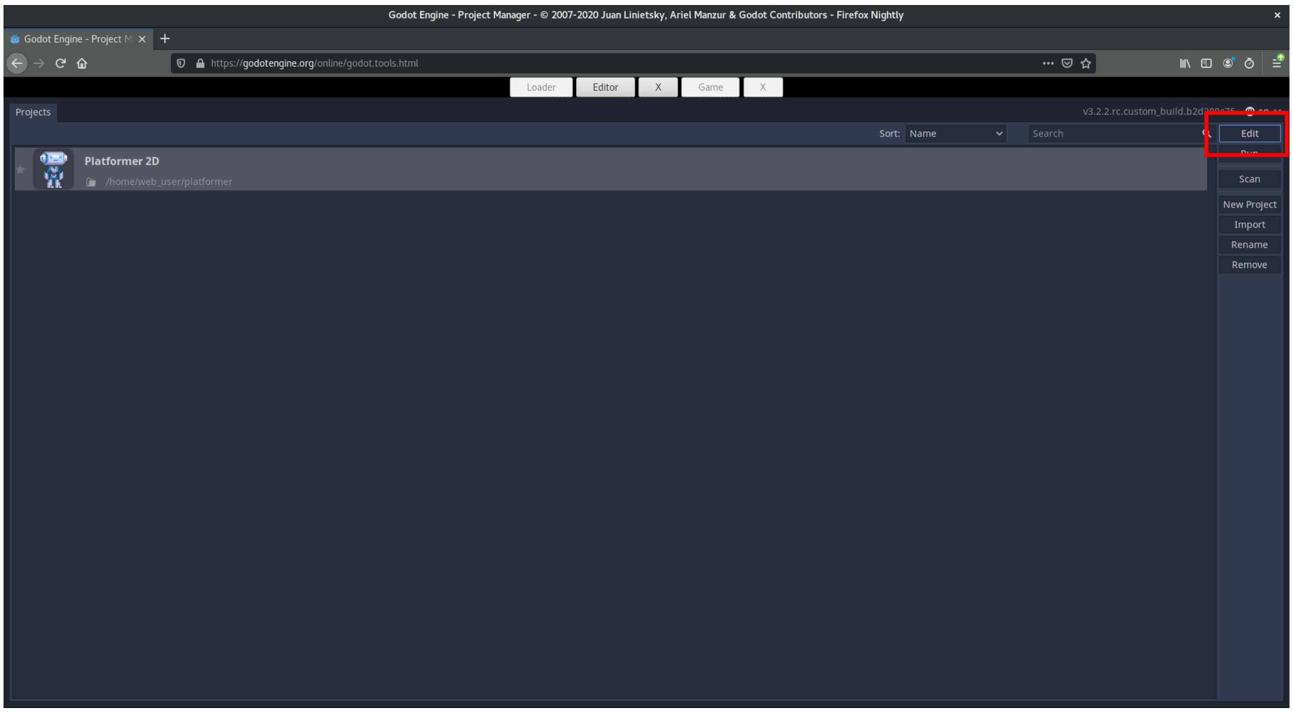
You can also opt to preload a zip file to the chosen virtual file system, allowing you to quickly test demo projects and load your offline projects inside the editor browser. Once the project manager starts, you will need to scan the virtual file system for new projects via the “Scan” option when preloading a zipped project. The editor config with the available projects list and other options will be stored according to your persistence method.

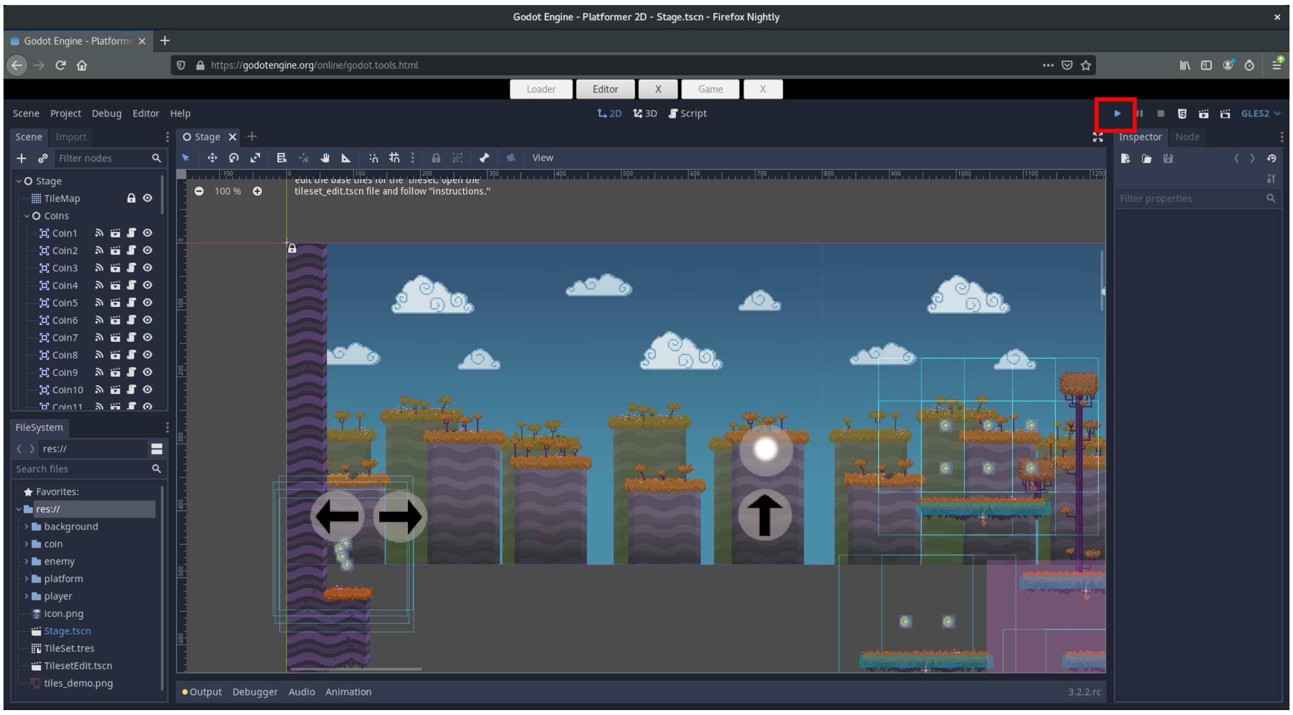


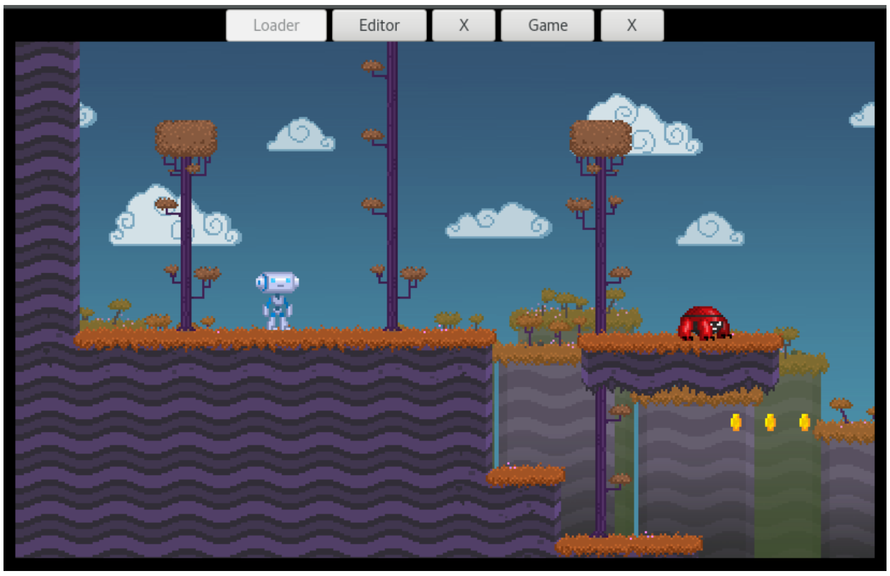


Editing & Running the project.

Once you have imported a project, or after creating a new one, you will be able to edit it, create new scenes, create new scripts, and upload assets via drag and drop. You will also be able to run your editor project inside the editor via the play button. Extra HTML UI allows you to close the game and the editor, and switch between them.


Godot Engine - Project Manager - © 2007-2020 Juan Linietsky, Ariel Manzur & Godot Contributors - Firefox Nightly
https://godotengine.org/online/godot.tools.html
Loader
Editor
X
Game
X
Projects
Sort: Name
Search
Platformer 2D
/home/web_user/platformer
Edit
Scan
New Project
Import
Rename
Remove





Known limitations

As stated above, this is a very early stage prototype, and there are some known limitations:

- Importing and using audio assets is still not well supported (and may cause a deadlock prompting the browser to ask the user to stop the script).

- Closing the project manager and game the via the HTML UI works, but closing the editor might deadlock and will always cause a memory leak.

- Sometimes refreshing the page when an error occurs is not enough, this is mostly due to browsers not clearing WebAssembly memory correctly. You might find yourself in a situation (after many reloads usually) where you have to open a new browser window/tab and visit the URL again for the editor to be able to run.

- The debugger connection between the editor instance and the running game does not work currently, so `print()` output or errors will not be raised in the editor. They are however accessible from the browser console.

- SharedArrayBuffer support is still limited among browsers. Recent versions of Chrome will work, as well as Mozilla Firefox Nightly builds (the beta and stable versions don't have SharedArrayBuffer enabled yet). Other browsers are untested.