

Make a copy of this checklist to your drive, and use it to keep track of your progress when implementing all the new features in your own source port.

Just fill the left-most cell with any value to indicate that it is completed.

General features and miscellaneous work

Music audio format - MP3

Music audio format - OGG Vorbis

Music audio format - Tracker

Allow for categorisation of features via an enumeration that matches the feature level definition

Either adapt or create new V_DrawPatch routines that map all calls to a virtual 320x200 space

Allow V_DrawPatch routines to accept transparency maps and translation tables

Update the 3D renderer to allow colormaps to be defined per-sector, overriding the Boom transfer colormap when it is set

Update the flat renderer to allow rotations

Update r_segs to render skies via user-defined skies as well as the original F_SKY hardcoded sky flat

Update the mobjinfo, state, sound, sprite, weapon, and ammo structures with the new defined fields

Copy the mobjinfo, state, sound, sprite, weapon, and ammo tables and adapt to your codebase

Update existing mobjinfo with new defaults and newly defined values for exceptional mobjinfo types

Adapt the mobjinfo, state, sound, sprite, weapon, and ammo table variables to be associative maps

Add an associative map for the doomednum field in a mobjinfo

Construct the associative maps in the manner described in the DeHackEd spec

Change the map thing spawner to ignore 0 and -1, spawn players between 1 and 4, and lookup the doomednum associative map for a valid mobjinfo when not a deathmatch start (including negative indices)

Update the weapon selection code to select by slot instead of by the hardcoded weapon enumeration when this feature level is enabled

Update the player's weaponowned, ammo, and maxammo fields to allow arbitrary weapon and ammo indices

Update the item pickup code in p_inter to use the new mobjinfo values when this feature level is enabled and when it is not one of the vanilla sprites set on the object

Update the respawning monsters code to use the new mobjinfo values

Update the action function structure to verify the type of function being set; and that the parameters the function is being called with match the set function signature

Change am_noammo and wp_nochange to equal -1

JSON Lumps

Import or write a JSON parser

Write a generic handler for JSON lumps that takes a function to execute with the "data" element as a parameter

Interlevel lumps

Add the exitanim and enteranim fields to UMAPINFO

Write a JSON parser for interlevel lumps

Adapt the wi_stuff file to allow data-driven creation of elements

Create a conditions handler

Ensure previous behavior is retained when no interlevel lumps are in use

Finale lumps

Add the endfinale fields to UMAPINFO

Write a JSON parser for finale lumps

Adapt the f_finale files to allow data-driven finales (music, backgrounds, bunny overlay behavior, cast roll calls)

Allow progression out of a finale if you can go to a new map

Create a new animation-based cast roll call as an alternative to the hardcoded roll call

	Ensure previous behavior is retained when no finale lumps are in use
Status bars	
	Write a JSON parser for the SBARDEF lump
	Adapt the st_lib and st_stuff files to allow data-driven creation of elements
	Create a conditions handler
	Create a generic number font renderer
	Create an animation element
	Create a hierarchical renderer and allow elements to have any number of children
	Ensure the screenblocks value can select all loaded status bars; clamp at the previous minimum and maximum values otherwise
Translations	
	Write a JSON parser for translation lumps
	Add hardcoded translation entries for T_GREEN, T_INDIGO, T_BROWN, T_RED, T_YELLOW, T_BLUE, T_NAVY, and T_MAGENTA and allow them to be overridden by WAD entries with the same name
	Ensure there is a path for the 3D renderer to accept arbitrary translation tables from mobjs
Skies	
	Write a JSON parser for the SKYDEF lump
	Implement the Doom 64/Playstation Doom firesky using any texture defined in TEXTURE1/2 as a target
	Implement Hexen-style transparency (0 palette index is completely transparent) for layered skies
	Update the sky renderer to resolve skies based on flat name, and allow rendering a foreground sky texture with Hexen-style transparency
Demo loop	
	Write a JSON parser for DEMOLOOP lump
	Update the demo loop code in d_main to defer to a demoloop structure when defined
Game configuration	
	Write a JSON parser for the GAMECONF lump
	Add functionality to the wad handler to allow unloading WADs
	Parse all specified IWAD and PWAD files for GAMECONF lumps, then unload all WADs
	Load all collected WADs from the GAMECONF pass
	Implement the new compatibility flags for the OPTIONS lump
Mapping additions	
	Create a new exit type that resets player inventory
	Create a music changer that is immediate and resolves lumps by exact name
	Update texture scrollers to allow scrolling front and back sidedefs
	Transfer sector colormaps when transferring floor and ceiling lighting
	Add the new line types to the linedef activation handlers
	Update UMAPINFO's bossaction field to allow thing numbers as well as mnemonics; and add the bossactionednum field to use a thing's doomednum instead
DeHackEd additions	
	Change allocation of new objects to push to the back of the in-order table and insert in to the associative maps
	Allow weapon and ammo definitions to allocate new weapons and ammo
	Check for and allocate objects whenever a frame, sound, sprite, mobjinfo, weapon, or ammo reference is encountered
	Add the new hardcoded strings to your string tables

	Allow user-defined string mnemonics starting with USER_ to be added to your string lookup table
	Add each new field defined for each type to your DeHackEd parser
	Reconstruct the associative maps after each patch is parsed
	Implement the FNV-1a algorithm
	Hash each item in each in-order table and compare against provided hashes in the next loaded DeHackEd patch