Crate Factory

Technical Documentation

This manual is for those who would like to create new material sets for the parts of the Crate Factory product either for personal or commercial.

First some words of wisdom for anyone who want to do something extra with Crate Factory, there are limitations, and they are bound to the actual props that comes with the set. There is no idea trying to create a new create with different dimensions, it will fail as the cf_cratemaker script is tuned to work with objects of the dimensions given in the set, nothing else.

But what is possible to do is:

- Create new crate sides, beams, nuts and bolts or skids, just assure they do load at the position and rotation of the props in the product.

- Create new material sets for existing prop parts, remember that every material has a low resolution counterpart and a low resolution texture map set, which is crucial to the functionality of creating create sets that does now totally hog your system.

- Create new and interesting definitions using the existing props and materials to increase the difference in looks between crates, and create definitions to use any new props or materials you have created.

This technical manual will only focus on that last bit, the anatomy if the definition files.

Important note!
When editing these files, it is very important to use a real text editor that does to have the horrible idea of replacing quotes with typographic quotes. For JSON files, the quotes used are standard double quotes, but if they are replaced by typographic quotes, the file will not load.

The main functionality is driven by two types of files, the Crate Factory Model Definition files, .cfmd, and the Crate Factory Object Definition, .cfod.

The definition files are pure JSON files, you can learn more here: https://www.json.org

We will begin with the .cfmd file, as that file is what controls the whole design of a crate, and also holds the information visible to the user.

The header of the .cfmd file holds some basic information about the model definition, where most of it is used by the GUI in the Single Crate Creator or the Crate Factory to display information about the crate this definition will generate.

title: The title of the module definition shown to the user.

info: A short descriptive text describing the result and the options available, and also which settings this crate is best suited for.

thumb: The filename of the thumb file, located in private/thumbs and should be a transparent .png at 180 x 180 pixels. The thumb should show some variations of the resulting crate.

basesize: an object describing the prop frame in DAZ Studio units and also what type it is, square or rectangular. This information is only used as information to the cf_cratebuilder but do not rely on making your own prop will work just because you enter it's dimensions here. The members of this object are:
x: width in X-space
z: depth in Z-space
type: square or rect

settings: An array of strings listing every setting this crate would fit into. You can add any words here and it will be available in the filter in Single Crate Creator and in Crate Factory.

Below follows the objects that describe the actual prop definitions and parameters.

frame: The frame object array. In the frame array you can add multiple frame definition objects, which means that if there are two frames that do have the same size (within margins), you can randomly select one of those frames and the base frame for the crate. Each frame object has the following layout:

object: name of the .cfod definition file for this object.

attachments: either **null** (which means no attachments attached) or an array of optional attachments for the frame. Each attachment object has the following members.

type: A string code for the attachment used for identifying the type, for example all feet has the type **feet**, all joints have the type **joint**. This is so the cf_cratebuilder only picks one attachment of each type.

name: The name shown in the GUIs where this attachment can be enabled or disabled.
optional: If the attachment is optional and will have a checkbox in the GUI where it can be turned on or off.

optionalchance: [Optional] The chance that this texture option is activated. 1 - 100, if this setting is not given, the chance is 100 per default.

object: name of the .cfod definition file for this object.

chance: a figure from 1 - 100, percentage chance, that this is the picked attachment when several attachments are listed. If the sum of the total chances are less than 100, the last in the list will get the remaining chances.

sides: This is the sides object array, which holds an array of position objects that defines each of the sides, which object options and beam options that side will have.

position: The position describing each side. It holds two objects, one array of integers listing which sides this position handles, and **data**, which is an array of the actual definitions. The sides are numbered as follows:

1 = Front
2 = Back
3 = Right
4 = Left
5 = Top
6 = Bottom

data: An array definition objects of a position that holds information for that position. When randomly picking, the same .cfod file will be used for all the sides controlled by the position object.

Example:
If you define a position for sides 1 - 4, containing two different definitions, one with a steel side and one with a plywood side, all sides 1 - 4 will be either steel or plywood, but the .cfod file will give the material options to use for each side. Then you have another position for sides 5 and 6 that only lists a wooden plank side, then that .cfod file will give the material options for those.

object: name of the .cfod definition file for this object.

chance: [Optional] a figure from 1 - 100, percentage chance, that this is the picked side when several sides are listed. If the sum of the total chances are less than 100, the last in the list will get the remaining chances.

type: [Optional] A string identifying this teature option, which is the option for a special texture option on the first of the listed sides.

name: The name shown in the GUIs where this option can be enabled or disabled.

optional: If this texture option is optional and will have a checkbox in the GUI where it can be turned on or off.

optionalchance: [Optional] The chance that this texture option is activated. 1 - 100, if this setting is not given, the chance is 100 per default.

beams: Either **null**, or an array with possible beam objects to be attached to this side.

object: name of the .cfod definition file for this object.

chance: a figure from 1 - 100, percentage chance, that this is the picked beam when several beams are listed. If the sum of the total chances are less than 100, the last in the list will get the remaining chances.

type: [Optional] A string identifying this beam option.

name: The name shown in the GUIs where this option can be enabled or disabled.

level: [Optional Sets at which detail level the bean is loaded or not. The detail levels are:

1 = High details
2 = Medium details
3 = Low details

This setting is particularly interesting if you create a very detailed beam that you do not want to be available on lower detail levels.

attachments: either **null** (which means no attachments attached) or an array of optional attachments for the beam. The attachment object is as described above.

As you see, there are a lot of flexibility built into the Crate Factory Model Definition files. Now, let's take a look at the Crate Factory Object Definition files too see how they interact with the Model Definitions. The header will give some important base information for the cf_cratebuilder script on how and where to load the prop,.

type: What type of object this is, the options are: **Side**, **Frame**, **Beam** or **Attachment**. The cf_cratemaker will cowardly refuse to load a Object Definition of the wrong type.

class: A text describing the type of Object this file will load. This is descriptive information only.

directory: In which subdirectory of Crate Maker the object and materials will be found. This is very important. By default there are two base directories, **Parts Metal** and **Parts Wood**.

object: The name of the actual Prop that should be loaded, just the name, not any path information, as the prop is loaded through the DAZ Studio loading mechanism.

mats: This is an array of Material definition objects of four different types. The four types and their usage will be described below.

mat: Holds the name to the actual Material definition file, just the file, not any path information. You add one mat object for each optional texture you have for this object. This is as objects with detail level 1, high, will have different textures on each side, textures that are from the same base but with a different touch, to make the crate look less repetitive when several are stacked together. On level 2, medium level, one randomly picked texture will be used on all sides.

matlow: Holds the name to the low resolution Material definition file, which will be used at a detail level 3. The texture maps used should be smaller, about 25% of the original size, as objects on level 3 will be away from the camera and smaller.

matfirst: This is either an array, or a single file name of a material file. This material is used if type, name and option is set in a data object for a side position in the .cfmd file that load this .cfod file. This is the trick to have some side textures with text, logos or signs on.

cf_cratemaker will when the crate creation is started, choose one of these options for the crate, and then if the matfirst option is an array randomly select diffent materials of the same type for different sides, again to avoid repetitive look, or if it is only a material definition file name, use that for every first side.

The first side is the side that comes first in the position side array in the .cfmd file.

Example:
If the .cfmd file has three position entries for the side object, one saying 1,2,3, one saying 4, and one saying 5,6, the first sides are 1, 4 & 5, which means that side 1, 4 & 5 will use the matfirst material.

matfirstlow: Exactly the same thing as **matfirst**, but the low resolution version, works the same way.

So, that was the technical walkthrough of the parameters in the definitions files. There are more files in the private directory but those are not for editing, but for the predefined layouts and not interesting to edit, I promise. Now it's time to look at some files and see how it looks.

Let's take a look at the .cfmd file for the Modern Plywood Crate, a little shortened.

```
{
"title": "Modern Plywood Crate",
"info": "A crate with a wooden frame, plywood sides, skids and bolts and occasional beams, with optional text on the top and on one side, suitable for modern or post apocalyptic settings. The crate has two slightly different frame options randomly picked, and it has user options to choose with or without wood skids.",
"thumb": "mc_plywood.png",
"basesize": { "x": 110, "z": 110, "type": "square" },
"settings": ["Modern", "Post apocalyptic"],

"frame": [{
        "object": "woodframe1modern.cfod",
        "attachments": [{
                "type": "skids",
                "name" : "Wood Skids",
                "optional" : true,
                "object": "skidswood.cfod",
                "chance": 100
        },
        {
                "type": "framejoints",
                "name" : "Frame Joints",
                "object": "metaljointwframe1.cfod",
                "level": 1,
                "chance": 100
        }]
}],
```

First, always, a JSON file starts with a {, and ends with a }, which you will see when we come to the end of the file. Here you first see title, info, thumb, basesize & settings. title info & thumb are defined as strings, while basesize is an object with three members and settings is an array with two string values.

Now we look at the frame object, as you see it's one frame, the woodframe1modern.cfod with two diffrerent attachments, skidswood.cfod and metaljointwframe1.cfod. The framejoints attachment is only loaded when the level is 1, highly detailed, as they are barely visible on longer distances than close.

The skids are optional, and will show a checkbox in the GUI in both in the Single Crate and Crate Factory scripts.

You could add an optionalchance here, which would randomly set or nor set the skids, as this works for all attachments.

```
"sides": [
{ "position": [1, 2, 3, 4], "data":
       [{
       "object": "woodsideplywoodtext.cfod",
       "chance": 100,
       "type": "sidetext",
       "name" : "Text on side",
       "optional": true,
       "optionalchance": 65,
       "beams": [{
              "object": "woodbeammodern1.cfod",
              "chance": 20,
              "name": "Wood Beams",
              "level": 3,
              "attachments":  [{
              "type": "beambolts",
              "name" : "Beam Bolt",
              "object": "beambolts1modern.cfod",
              "level": 1,
              "chance": 100
              }]
       },{
              "object": "woodbeammodern2.cfod",
              "chance": 20,
              "name": "Wood Beams",
              "level": 3,
              "attachments":  [{
              "type": "beambolts",
              "name" : "Beam Bolt",
              "object": "beambolts2modern.cfod",
              "level": 1,
              "chance": 100
              }]
       }}]
}]},
{ "position": [5,6], "data":
       [{
              "object": "woodsideplywoodtextup.
cfod",
              "type": "toptext",
              "name" : "Text on top",
              "optional": true,
              "optionalchance": 65,
              "chance": 100,
              "beams": null
       }]
}]}
```

Now we look at the sides definition. As you see it has two position blocks, one for sides 1-4 (the sides), and one for 5-6, top and bottom.

object tells which .cfod file to be read for prop and material options, in this case woodsideplywoodtext.cfod, then beams defines which beam options exists, and what attachments these beams will have.

optionchance of the side tells cf_cratemaker that there is a 65% chance that the side should have the matfirst/matlowfirst texture, which is the optional texture with text or signs on.

There are two different beams, either woodbeammodern1.cfod or woodbeammodern2.cfod to choose from, (in the real file there are three, but one was omitted due to space considerations for this manual.

As you see the beams will be available up to level 3 (low resolution), while the beam bolt attachments will only be available on level 1 (high resolution).

On sides 5 and 6, there are no beams, beam says null. It's important to set beams to null, and not just omit it, as cf_cratemaker will treat a cfmd file without beams as broken.

```
{
"type": "Side",
"class": "Metal Steel",
"directory": "Parts Metal",
"object": "CF Flatside 2.duf",
"mats": [
{ "mat": "CF Flatside 2 Steel 1.duf" },
{ "mat": "CF Flatside 2 Steel 2.duf" },
{ "mat": "CF Flatside 2 Steel 3.duf" },
{ "mat": "CF Flatside 2 Steel 4.duf" },
{ "mat": "CF Flatside 2 Steel 5.duf" },
{ "mat": "CF Flatside 2 Steel 6.duf" },
{ "mat": "CF Flatside 2 Steel 7.duf" },
{ "mat": "CF Flatside 2 Steel 8.duf" },

{"matfirst": [
"CF Flatside 2 Steel Toxic 1.duf",
"CF Flatside 2 Steel Toxic 2.duf"]},
{"matfirst": [
"CF Flatside 2 Steel Nuke 1.duf",
"CF Flatside 2 Steel Nuke 2.duf"]},
{"matfirst": [
"CF Flatside 2 Steel VX 1.duf",
"CF Flatside 2 Steel VX 2.duf"]},

{ "matlow": "CF Flatside 2 Steel 1_low.duf" },
{ "matlow": "CF Flatside 2 Steel 2_low.duf" },
{ "matlow": "CF Flatside 2 Steel 3_low.duf" },
{ "matlow": "CF Flatside 2 Steel 4_low.duf" },
{ "matlow": "CF Flatside 2 Steel 5_low.duf" },
{ "matlow": "CF Flatside 2 Steel 6_low.duf" },
{ "matlow": "CF Flatside 2 Steel 7_low.duf" },
{ "matlow": "CF Flatside 2 Steel 8_low.duf" },

{"matlowfirst": [
"CF Flatside 2 Steel Toxic 1_low.duf",
"CF Flatside 2 Steel Toxic 2_low.duf"]},
{"matlowfirst": [
"CF Flatside 2 Steel Nuke 1_low.duf",
"CF Flatside 2 Steel Nuke 2_low.duf"]},
{"matlowfirst": [
"CF Flatside 2 Steel VX 1_low.duf",
"CF Flatside 2 Steel VX 2_low.duf"]}
]
}
```

Let ut take a look at a complicated .cfod file, the steel-side.cfod.

As you see first comes a header that says this is a Side, and it's Metal Steel, and it should load the prop CF Flatside 2.duf from the directory Parts Metal, which is a subdirectory of the product directory Crate Factory.

The he have a series of mats, first the normal mat ones, telling us there are 8 materiels variatrions to the CF Flatside 2 Steel, 1 - 8, and the same for the lowres versions, under the matlow object.

The interesting parts here are the matfirst definitions (and also the matlowfirst, but they are basically identical in functionality to the matfirst, just points to the lowres materials.

the matfirst object contains an array here, with two different materielsm the first matfirst says "CF flatside 2 Steel Toxic 1.duf" and the second says "CF flatside 2 Steel Toxic 2.duf". That tells us that the material CF flatside 2 Steel Toxic has two versions of if, the same basic texture with differences in it, in this case different dirt and wear, this is to make the crates looks even more differentiated from each other even when they have the same basic texture.

If there only is one variation in the matfirst, no array is used and it's entered just as a normal mat object.

The reason for this is that the first time cf_cratemaker finds a matfirst or a matlowfirst object it randomizes between them, and keeps the outcome and reuses the number for any subsequent randoms, so id you have one set of textures with one text and signs and another with another text and sign, it will keep "on the same theme" so to speak.

I hope this manual dod not bore you t death, and remember, it was written for those who would like to expand Crate Factory with their own definitions.