C++ API

C4DtoA is shipped with an API for 3rd party developers to write modules (basically custom translators).

The API is a static library located in the **$C4D_INSTALL/plugins/C4DtoA/api** folder. To add it to your project you have to add the following settings:

- header search path: **$C4D_INSTALL/plugins/C4DtoA/api/include**
- library search path: **$C4D_INSTALL/plugins/C4DtoA/api/lib**
- input library: **c4dtoa_api.lib**
- The Arnold dll is delay loaded on **Windows** therefore you have to specify the ***/DELAYLOAD:ai.dll*** linker flag.

In the code just simply include the *c4dtoa_api* header.

```
#include "c4dtoa_api.h"
```

If you want to use any C4DtoA node or parameter ids:

1. Add the following paths to your header search path:
    - **$C4D_INSTALL/plugins/C4DtoA/res/description**
    - **$C4D_INSTALL/plugins/C4DtoA/res/dialog**
2. Include **c4dtoa_symbols.h** or any description or dialog resource header file (e.g. ainode_options.h).

> ⚠ The C4DtoA module does not work properly if it's located in a subfolder named *c4dtoa* (e.g. */Applications/MAXON/CINEMA 4D R18/plugins /myplugin/c4dtoa/mymodule.dylib*).

UI framework

C4DtoA has its own UI generator framework which utilizes Arnold's node structure and the Arnold Metadata API. This means widgets of Arnold node parameters are generated automatically by the type and metadata of the parameters and parameter values are automatically exported.

Parameter ids

Each parameter has a unique id which is generated from the node entry name and parameter name. To generate an id use the **paramid_generator** tool of the API. For example, run the following command to get the id of the *Kd_color* parameter of the *standard* node:

```
$C4D_INSTALL/plugins/C4DtoA/api/bin/paramid_generator standard.Kd_color
```

Resource file

In the .res file use the AIPARAM data type. This tells C4DtoA that this parameter belongs to the Arnold node so the framework will generate the required widget in this position. Each parameter type has its own widget, some custom widget can be defined in metadata files (.mtd), see Metadata file (.mtd).

```
AIPARAM C4DAIP_STANDARD_KD_COLOR {}
```

Parameters which are not listed in the resource file will be added to a tab called *"Unsorted"* until you did not hide them directly in the metadata file.

NOTE: 3rd party custom widgets cannot be added via the API at the moment.

Labels

Labels are generated from the name of the parameter by default. Sometimes we want custom labels. For example, we want to display Diffuse color instead of Kd color. In this case, we have to define the label in the .str file just like by a normal C4D parameter.

```
C4DAIP_STANDARD_KD_COLOR    "Diffuse color";
```

Metadata file (.mtd)

Some special flags which used by the framework can be defined in the metadata file (.mtd).

Node flags:

- **c4d.classification (STRING)**: available for shaders to define their type. Available values are:
    - generic (default): the shader can be used in any contexts.
    - surface: shades the surface of a shape.
    - environment: defines a scene background (*options.background*).
    - atmosphere: defines light scattering effect in the scene (*options.atmosphere*).
    - light_filter: modifies output of a light source.
    - displacement: defines displacement effect of a polymesh.
    - texture: image or procedural texture shader.
    - volume: shades a volumetric object.
- **c4d.menu (STRING)**: available for shaders to define their path in the Material Browser menu.

- **c4d.command_id (INT)**: unique id obtained from Plugin Café. Needed for the command to create the Material node of the Arnold shader.

Parameter flags:

- **default (INT/FLOAT/STRING)**: overrides default value of the parameter.
- **min (INT/FLOAT)**: minimum value of the parameter. If both min/softmin and max/sotfmax is defined then a slider widget is rendered.
- **max (INT/FLOAT)**: maximum value of the parameter. If both min/softmin and max/softmax is defined then a slider widget is rendered.
- **softmin (INT/FLOAT)**: minimum value of the slider. User can add lower values (down to min) manually.
- **softmax (INT/FLOAT)**: maximum value of the slider. User can add higher values (up to max) manually.
- **c4d.step (INT/FLOAT)**: step size of the widget (using the slider or edit arrows).
- **c4d.gui_visible (BOOL)**: if false the parameter will not be visible on the GUI and will not be exported (default is true).
- **c4d.tag_visible (BOOL)**: if false the parameter will not be visible on the Arnold Parameters tag (default is true).
- **c4d.exportable (BOOL)**: if false the parameter will not be exported (default is true).
- **c4d.gui_type (INT)**: custom GUI types
  - 0: filename (file) - available by string parameter
  - 1: filename (dir) - available by string parameter
  - 2: filename (save) - available by string parameter
  - 3: filename (texture) - available by string parameter
  - 4: shader link - available by node parameter
  - 5: original field with shader link
  - 6: weight (a float field with 0.0 - 1.0 interval) - available by float parameter
  - 7: percent unit - available by int and float parameter
  - 8: curve editor - available by float array parameter
  - 9: visibility - available by int parameter
  - 10: meter unit - available by int and float parameter
- **c4d.array_type (INT)**: defines how to export an array type parameter
  - 0: array parameter: normal array, values are translated in step 0 (default).
  - 1: motion blur array: scalar value is translated in each motion step.
  - 2: motion blur array parameter: array values are translated in each motion step.

An example definition:

```
[node standard]
 c4d.classification      STRING  "surface"
 c4d.menu                STRING  "shader/surface"
 c4d.command_id          INT     1031853

 [attr Kd]
     c4d.gui_type        INT     6

 [attr direct_diffuse]
     min                 FLOAT   0.0
     softmax             FLOAT   1.0
     c4d.step            FLOAT   0.01
```