

---

# **Celaria Map Toolkit**

*Release 0.2.0*

**Iceflower S**

**Aug 24, 2019**



# CONTENTS

<b>1</b>	<b>CMT - Celaria Map Toolkit</b>	<b>1</b>
1.1	About . . . . .	1
1.2	Web . . . . .	1
1.3	Credits . . . . .	2
	<b>Python Module Index</b>	<b>23</b>
	<b>Index</b>	<b>25</b>



## CMT - CELARIA MAP TOOLKIT

---

Celaria Map Toolkit can convert different map format from one into another.

Install via pip:

```
pip install cmt
```

About the usage see:

```
cmt -help
```

---

## 1.1 About

### 1.1.1 .cmap support

Version	Encode	Decode	Convert	Downgrade	Upgrade
0	✓	✓	✓		✓
1	✓	✓	✓	✓	

### 1.1.2 .ecmap support

Version	Encode	Decode	Convert	Downgrade	Upgrade
0	✓	✓	✓		✓
1	✓	✓	✓	✓	

---

## 1.2 Web

<https://github.com/IceflowRE/cmt>

---

## 1.3 Credits

- **Developer**
  - Iceflower S
    - \* iceflower@gmx.de
- **Format Definition**
  - <https://www.celaria.com/>

### 1.3.1 Third Party

#### pytest

- Holger Krekel and others
- <https://github.com/pytest-dev/pytest>
- MIT

#### Prospector

- landscapeio
- <https://github.com/landscapeio/prospector>
- GPL-2.0+

#### Read the Docs Sphinx Theme

- Dave Snider
- [https://github.com/rtfd/sphinx\\_rtd\\_theme](https://github.com/rtfd/sphinx_rtd_theme)
- MIT

#### Setuptools

- Jason R Coombs / Setuptools Developers
- <https://github.com/pypa/setuptools>
- MIT

#### Sphinx

- the Sphinx team
- <https://github.com/sphinx-doc/sphinx>
- BSD-2-Clause

#### sphinx-autodoc-typehints

- Alex Grönholm
- <https://github.com/agronholm/sphinx-autodoc-typehints>
- MIT

#### twine

- various authors
- <https://github.com/pypa/twine>

- Apache-2.0

#### wheel

- Charlie Denton
- <https://github.com/meshy/pythonwheels>
- BSD-2-Clause

### 1.3.2 Disclaimer

This software is not official supported by <https://www.celaria.com/>.

### 1.3.3 License

Copyright 2019-present Iceflower S ([iceflower@gmx.de](mailto:iceflower@gmx.de))

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## User Guide

### Installation

#### System Requirements

Python is required in the version 3.7. or higher, it can be downloaded at <https://www.python.org/downloads/>.

During the Windows installation you should make sure that the `PATH` / environment variable is set and `pip` is installed.

Under Linux it should be ensured that `pip` is installed, if this is not done with the standard installation.

#### Installation

```
pip install cmt
```

### Using cmt

The program is a terminal program, so it runs from the terminal.

Calling with: .. code-block:: none

`cmt`

**-h, --help**

show this help message and exit

**-v, --version**

show program's version number and exit

**convert** file {cmap,ecmap} {0,1} output

convert file to type, version and output file

### Internals

Reference material.

#### cmt

#### cmt.a\_converter

**class** `cmt.a_converter.AConverter`

Bases: `abc.ABC`

**abstract static convert\_to** (*source*, *target*)

Convert to the other map format of same version.

**Return type** *AMap*

**abstract static downgrade** (*source*)

Downgrade to the format version below.

**Return type** *AMap*

**abstract static upgrade** (*source*)

Upgrade to the format version above.

**Return type** *AMap*

#### cmt.a\_map

**class** `cmt.a_map.AMap` (*identifier*, *version*)

Bases: `abc.ABC`

**abstract classmethod decode** (*data*, *offset*, *debug=False*)

**Return type** *AMap*

**abstract encode** ()

**Return type** `bytearray`



```

class cmt.a_map.MapType
    Bases: enum.Enum

    An enumeration.

    CMAP = 'celaria_map'
    ECMAP = 'celaria_edi'

    from_str(text) = <function MapType.from_str>

```

### cmt.convert

```

cmt.convert.convert(source, version, target)

    Raises ValueError – something failed

    Return type Union[CMAP, CMap, EMap, EMap]

```

### cmt.decode

```

cmt.decode.decode(file, debug=False)

    Raises ValueError – something failed

    Return type Union[CMAP, CMap, EMap, EMap]

```

### cmt.encode

```

cmt.encode.encode(source, file)

```

### cmt.static\_data

Static and important data which is needed by the program and do not need any third party libraries. (this is important because it is used inside the setup.py.

```

cmt.static_data.AUTHOR = 'Iceflower S'
    author

cmt.static_data.AUTHOR_EMAIL = 'iceflower@gmx.de'
    author email

cmt.static_data.DESCRPTION = 'Celaria Map Toolkit, can convert different map formats from
    short description

cmt.static_data.LONG_NAME = 'Celaria Map Toolkit'
    long name of this program

cmt.static_data.NAME = 'CMT'
    name of this program

cmt.static_data.PROJECT_URL = 'https://github.com/IceflowRE/cmt'
    project url

cmt.static_data.VERSION = '0.3.0.dev1'
    version in PEP440 format

```

## cmt.utils

**class** `cmt.utils.DebugIterUnpack` (*format\_, buffer, what*)  
Bases: `object`

`cmt.utils.debug_print` (*data, what, value, offset=None*)

`cmt.utils.to_hex` (*data*)

`cmt.utils.unpack_from` (*format\_, buffer, offset, what, debug*)  
Same behaviour as `struct.unpack_from`.

### Parameters

- **format** –
- **buffer** (`bytes`) –
- **offset** (`int`) –
- **what** (`Tuple[str, ...]`) – tuple of message for every unpacked value
- **debug** (`bool`) – use debug mode

### Returns

## cmt.blender

### cmt.blender.v2\_80

#### cmt.blender.v2\_80.import\_menu

**class** `cmt.blender.v2_80.import_menu.ImportCMap` (*\*args, \*\*kwargs*)  
Bases: `bpy.types.Operator`, `bpy_extras.io_utils.ImportHelper`

Import menu for `.cmap` or `.ecmap` file.

**bl\_idname** = `'import_scene.cmap'`

**bl\_label** = `'Import Celaria Map (.cmap/.ecmap)'`

**execute** (*context*)

Return type `Set[str]`

**filename\_ext** = `'.cmap;.ecmap'`

`cmt.blender.v2_80.import_menu.import_cmap` (*ops, context, filepath*)

`cmt.blender.v2_80.import_menu.menu_func_import` (*self, context*)

`cmt.blender.v2_80.import_menu.register` ()

`cmt.blender.v2_80.import_menu.unregister` ()

#### cmt.blender.v2\_80.object\_panel

**class** `cmt.blender.v2_80.object_panel.BlockProps` (*\*args, \*\*kwargs*)  
Bases: `bpy.types.PropertyGroup`

```

class cmt.blender.v2_80.object_panel.Dummy (*args, **kwargs)
    Bases: bpy.types.PropertyGroup

class cmt.blender.v2_80.object_panel.ObjectPanel (*args, **kwargs)
    Bases: bpy.types.Panel

    Creates a Panel in the scene context of the properties editor

    bl_context = 'object'
    bl_idname = 'OBJECT_PT_cmt'
    bl_label = 'Celaria Object'
    bl_region_type = 'WINDOW'
    bl_space_type = 'PROPERTIES'

    draw (context)

    draw_header (context)

    classmethod poll (context)

cmt.blender.v2_80.object_panel.register (version=1)
cmt.blender.v2_80.object_panel.unregister ()

```

### **cmt.blender.v2\_80.scene\_panel**

```

class cmt.blender.v2_80.scene_panel.MetadataProps (*args, **kwargs)
    Bases: bpy.types.PropertyGroup

class cmt.blender.v2_80.scene_panel.ScenePanel (*args, **kwargs)
    Bases: bpy.types.Panel

    Creates a Panel in the scene context of the properties editor

    bl_context = 'scene'
    bl_idname = 'SCENE_PT_layout'
    bl_label = 'Celaria Metadata'
    bl_region_type = 'WINDOW'
    bl_space_type = 'PROPERTIES'

    draw (context)

    draw_header (context)

cmt.blender.v2_80.scene_panel.register ()
cmt.blender.v2_80.scene_panel.unregister ()

```

### **cmt.blender.v2\_80.utils**

```

cmt.blender.v2_80.utils.add_exclusive_to_collection (obj, coll_name)

cmt.blender.v2_80.utils.create_material (name, color)
    Creates a material with a specific color, if the material name does not already exist. :type name: str :param
    name: :type color: Tuple[float, float, float, float] :param color: :rtype: bpy.types.Material :return:

```

`cmt.blender.v2_80.utils.get_collection` (*name*)  
If the collection is not existing it will create a new. :type name: `str` :param name: :rtype: `bpy.types.Collection`  
:return:

`cmt.blender.v2_80.utils.to_bl_location` (*location*)  
**Return type** `Tuple[float, float, float]`

`cmt.blender.v2_80.utils.to_bl_rotation_z` (*rotation\_z*)  
**Return type** `float`

`cmt.blender.v2_80.utils.to_bl_scale` (*scale*)  
**Return type** `Tuple[float, float, float]`

`cmt.blender.v2_80.utils.to_cmt_position` (*location*)  
**Return type** `Tuple[float, float, float]`

`cmt.blender.v2_80.utils.to_cmt_rotation_z` (*rotation\_z*)  
**Return type** `float`

`cmt.blender.v2_80.utils.to_cmt_scale` (*scale*)  
**Return type** `Tuple[float, float, float]`

### `cmt.blender.v2_80.viewport_add_menu`

**class** `cmt.blender.v2_80.viewport_add_menu.AddNewBlock` (*\*args, \*\*kwargs*)  
Bases: `bpy.types.Operator`  
Add a new Block  
**bl\_idname** = `'cmt.add_new_block'`  
**bl\_label** = `'Add a new Celaria Block'`  
**execute** (*context*)  
**Return type** `Set[str]`

**class** `cmt.blender.v2_80.viewport_add_menu.AddNewDummy` (*\*args, \*\*kwargs*)  
Bases: `bpy.types.Operator`  
Add a new Dummy  
**bl\_idname** = `'cmt.add_new_dummy'`  
**bl\_label** = `'Add a new Celaria Dummy'`  
**execute** (*context*)  
**Return type** `Set[str]`

**class** `cmt.blender.v2_80.viewport_add_menu.AddNewPlayerStart` (*\*args, \*\*kwargs*)  
Bases: `bpy.types.Operator`  
Add a new Player Start  
**bl\_idname** = `'cmt.add_new_playerstart'`  
**bl\_label** = `'Add a new Celaria Player Start'`  
**execute** (*context*)

**Return type** Set[str]

```
class cmt.blender.v2_80.viewport_add_menu.AddNewSphere (*args, **kwargs)
    Bases: bpy.types.Operator
```

Add a new Sphere

```
bl_idname = 'cmt.add_new_sphere'
```

```
bl_label = 'Add a new Celaria Sphere'
```

```
execute (context)
```

**Return type** Set[str]

```
class cmt.blender.v2_80.viewport_add_menu.ViewportAddMenu (*args, **kwargs)
    Bases: bpy.types.Menu
```

Create new Celaria Objects

```
bl_idname = 'VIEW3D_MT_add_cmt'
```

```
bl_label = 'Celaria Objects'
```

```
draw (context)
```

```
cmt.blender.v2_80.viewport_add_menu.menu_add_object (self, context)
```

```
cmt.blender.v2_80.viewport_add_menu.register ()
```

```
cmt.blender.v2_80.viewport_add_menu.unregister ()
```

## **cmt.blender.v2\_80.v1**

### **cmt.blender.v2\_80.v1.add\_objects**

```
cmt.blender.v2_80.v1.add_objects.add_block (ent, time=None)
    Add a block to collection. :return Created object.
```

**Return type** bpy.types.Object

```
cmt.blender.v2_80.v1.add_objects.add_camera (position, look_at, preview_cam_set)
```

**Return type** bpy.types.Object

```
cmt.blender.v2_80.v1.add_objects.add_dummy (ent)
```

**Return type** bpy.types.Object

```
cmt.blender.v2_80.v1.add_objects.add_player_start (ent)
```

```
cmt.blender.v2_80.v1.add_objects.add_sphere (ent)
```

**Return type** bpy.types.Object

```
cmt.blender.v2_80.v1.add_objects.add_sun (angle, rotation_z)
```

**Return type** bpy.types.Object

```
cmt.blender.v2_80.v1.add_objects.get_color (typ)
```

Get the color of a specific block.

**Return type** Tuple[float, float, float, float]

## cmt.blender.v2\_80.v1.codec

cmt.blender.v2\_80.v1.codec.**decode** (*cmap*)

## cmt.cmap

### cmt.cmap.a\_cmap

**class** cmt.cmap.a\_cmap.**ACMap** (*version*)

Bases: *cmt.a\_map.AMap*

**abstract classmethod decode** (*data, offset, debug=False*)

Return type *ACMap*

**abstract encode** ()

Return type *bytearray*

### cmt.cmap.a\_entity

**class** cmt.cmap.a\_entity.**AEntity** (*type\_, byte\_size*)

Bases: *abc.ABC*

#### Variables

- **type** – entity type
- **byte\_size** – size in bytes the entity uses

**abstract classmethod decode** (*data, offset, debug=False*)

#### Parameters

- **data** (*bytes*) –
- **offset** (*int*) – without entity type byte
- **debug** (*bool*) –

Return type *AEntity*

**abstract encode** ()

Includes the entity type.

Return type *bytearray*

## cmt.cmap.v0

### cmt.cmap.v0.cmap

**class** cmt.cmap.v0.cmap.**CMap**

Bases: *cmt.cmap.a\_cmap.ACMap*

Celaria .cmap format (version 0)

#### Datatypes

Abbreviation	Type	Byte size
uByte	unsigned byte	1
uShort	unsigned short	2
uInt	unsigned int	4
sShort	signed short	2
sInt	signed int	4
f32	float	4
f64	double	8

### Description format

> <datatype> (<number of datatypes in sequence>) // <description>

or

> [<variable name>] : <datatype> (<number of datatypes in sequence>) // <description>

### Format

```

> uByte (11) // string identifier
> uByte (1) // version

> nameLen : uByte (1) // number of characters in map name
> uByte (nameLen) // map name as String

> uByte (1) - boolean, if the timer will be run in singleplayer

> uByte (1) // unused byte

> times : uByte (1) - number of checkpoint times (including medal time)

> uInt (times) // checkpoint times for platin
> uInt (times) // checkpoint times for gold
> uInt (times) // checkpoint times for silver
> uInt (times) // checkpoint times for bronze

> f32 (1) // sun rotation on Z axis
> f32 (1) // sun height expressed as an angle (between 0 and 90 degrees)

> f64 (1) // preview camera position x
> f64 (1) // preview camera position y
> f64 (1) // preview camera position z
> f64 (1) // preview camera look at position x
> f64 (1) // preview camera look at position y
> f64 (1) // preview camera look at position z

> entityNumber : uInt (1) // number of entities on the map

for entity in entityNumber {
  > entityType : uInt (1) // entityType

  switch(entityType) {
    case 0: // block
      > blockType : uByte (1) // blockType/color
      > uByte (1) // unused byte
      > sInt (1) // position x
  }
}

```

(continues on next page)

(continued from previous page)

```

> sInt (1) // position y
> uInt (1) // position z
> uInt (1) // scale x
> uInt (1) // scale y
> uInt (1) // scale z
> f32 (1) // rotation on Z axis

if (blockType == 5){ // checkpoint block
    > uByte (1) // checkpoint Number
}

case 1: // sphere
    > sInt (1) // position x
    > sInt (1) // position y
    > sInt (1) // position z

case 2: // player start
    > uByte (1) // unused byte
    > sInt (1) // position x
    > sInt (1) // position y
    > sInt (1) // position z
    > f32 (1) // rotation on Z axis

case 128: // dummy id
    > uByte (1) // ID
    > sInt (1) // position x
    > sInt (1) // position y
    > uInt (1) // position z
    > uInt (1) // scale x
    > uInt (1) // scale y
    > uInt (1) // scale z
    > f32 (1) // rotation on Z axis
}

```

**classmethod decode** (*data*, *offset*, *debug=False*)

**Return type** CMAP

**encode** ()

**Return type** bytearray

### cmt.cmap.v0.entities

**class** cmt.cmap.v0.entities.**Block**

Bases: *cmt.cmap.a\_entity.AEntity*

**classmethod decode** (*data*, *offset*, *debug=False*)

**Parameters**

- **data** (bytes) –
- **offset** (int) – without entity type byte
- **debug** (bool) –

**Return type** *Block*



**encode ()**  
Includes the entity type.

**Return type** `bytearray`

**class** `cmt.cmap.v0.entities.BlockType`

Bases: `enum.Enum`

An enumeration.

**CHECKPOINT** = 5

**FINISH** = 1

**ICE** = 4

**JUMP** = 2

**NOTHING** = 0

**SPEED** = 3

**class** `cmt.cmap.v0.entities.Dummy`

Bases: `cmt.cmap.a_entity.AEntity`

**classmethod** **decode** (*data, offset, debug=False*)

**Parameters**

- **data** (`bytes`) –
- **offset** (`int`) – without entity type byte
- **debug** (`bool`) –

**Return type** `Dummy`

**encode ()**  
Includes the entity type.

**Return type** `bytearray`

**class** `cmt.cmap.v0.entities.PlayerStart`

Bases: `cmt.cmap.a_entity.AEntity`

**classmethod** **decode** (*data, offset, debug=False*)

**Parameters**

- **data** (`bytes`) –
- **offset** (`int`) – without entity type byte
- **debug** (`bool`) –

**Return type** `PlayerStart`

**encode ()**  
Includes the entity type.

**Return type** `bytearray`

**class** `cmt.cmap.v0.entities.Sphere`

Bases: `cmt.cmap.a_entity.AEntity`

**classmethod** **decode** (*data, offset, debug=False*)

**Parameters**

- **data** (*bytes*) –
- **offset** (*int*) – without entity type byte
- **debug** (*bool*) –

**Return type** *Sphere*

**encode** ()

Includes the entity type.

**Return type** *bytearray*

### **cmt.cmap.v0.medal\_time**

**class** `cmt.cmap.v0.medal_time.MedalTime` (*platin=0, gold=0, silver=0, bronze=0*)

Bases: *object*

`cmt.cmap.v0.medal_time.decode_medal_times` (*data, offset, debug=False*)

Must start with the length byte.

#### **Parameters**

- **data** (*bytes*) –
- **offset** (*int*) –
- **debug** (*bool*) –

**Return type** *List[MedalTime]*

### **cmt.cmap.v1**

#### **cmt.cmap.v1.cmap**

**class** `cmt.cmap.v1.cmap.CMap`

Bases: *cmt.cmap.a\_cmap.ACMap*

Celaria .cmap format (version 1)

#### **Datatypes**

Abbreviation	Type	Byte size
uByte	unsigned byte	1
uShort	unsigned short	2
uInt	unsigned int	4
sShort	signed short	2
sInt	signed int	4
f32	float	4
f64	double	8

#### **Description format**

> <datatype> (<number of datatypes in sequence>) // <description>

or

> [*<variable name>*] : <datatype> (<number of datatypes in sequence>) // <description>

**Format**

```

> uByte (1) // string identifier
> uByte (1) // version

> nameLen : uByte (1) // number of characters in map name
> uByte (nameLen) // map name as String

> uByte (1) // boolean, previewCam_set

> uByte (1) // number of checkpoint times (including finish line)

> times : uByte (1) // number of checkpoint times (including finish line)

> uInt (times) // checkpoint times for platin
> uInt (times) // checkpoint times for gold
> uInt (times) // checkpoint times for silver
> uInt (times) // checkpoint times for bronze

> f32 (1) // sun rotation on Z axis
> f32 (1) // sun angle to xy plane (between 0 and 90 degrees)

> f64 (1) // preview camera position x
> f64 (1) // preview camera position y
> f64 (1) // preview camera position z
> f64 (1) // preview camera look at position x
> f64 (1) // preview camera look at position y
> f64 (1) // preview camera look at position z

> entityNumber : uInt (1) // number of entities on the map

for entity in entityNumber {
  > entityType : uInt (1) // entityType

  switch(entityType) {
    case 0: // block
      > blockType : uByte (1) // blockType/color
      > sInt (1) // position x
      > sInt (1) // position y
      > uInt (1) // position z
      > uInt (1) // scale x
      > uInt (1) // scale y
      > uInt (1) // scale z
      > f32 (1) // rotation on Z axis

      if (blockType == 5){ // checkpoint block
        > uByte (1) // checkpoint Number
      }

    case 1: // sphere
      > sInt (1) // position x
      > sInt (1) // position y
      > uInt (1) // position z

    case 2: // player start
      > uByte (1) // unknown
      > sInt (1) // position x
      > sInt (1) // position y

```

(continues on next page)

(continued from previous page)

```

        > uInt (1) // position z
        > f32 (1) // rotation on Z axis

    case 128: // dummy id
        > uByte (1) // ID
        > sInt (1) // position x
        > sInt (1) // position y
        > uInt (1) // position z
        > uInt (1) // scale x
        > uInt (1) // scale y
        > uInt (1) // scale z
        > f32 (1) // rotation on Z axis
    }
}

```

**classmethod** `decode` (*data*, *offset*, *debug=False*)

**Return type** `CMap`

**encode** ()

**Return type** `bytearray`

### **cmt.cmap.v1.entities**

**class** `cmt.cmap.v1.entities.Block`

Bases: `cmt.cmap.a_entity.AEntity`

**classmethod** `decode` (*data*, *offset*, *debug=False*)

**Parameters**

- **data** (`bytes`) –
- **offset** (`int`) – without entity type byte
- **debug** (`bool`) –

**Return type** `Block`

**encode** ()

Includes the entity type.

**Return type** `bytearray`

**class** `cmt.cmap.v1.entities.BlockType`

Bases: `enum.Enum`

An enumeration.

**CHECKPOINT** = 5

**FINISH** = 1

**ICE** = 4

**JUMP** = 2

**NOTHING** = 0

**SPEED** = 3

```
class cmt.cmap.v1.entities.Dummy
    Bases: cmt.cmap.a_entity.AEntity

    classmethod decode (data, offset, debug=False)
```

**Parameters**

- **data** (*bytes*) –
- **offset** (*int*) – without entity type byte
- **debug** (*bool*) –

**Return type** *Dummy*

```
encode ()
    Includes the entity type.
```

**Return type** *bytearray*

```
class cmt.cmap.v1.entities.PlayerStart
    Bases: cmt.cmap.a_entity.AEntity

    classmethod decode (data, offset, debug=False)
```

**Parameters**

- **data** (*bytes*) –
- **offset** (*int*) – without entity type byte
- **debug** (*bool*) –

**Return type** *PlayerStart*

```
encode ()
    Includes the entity type.
```

**Return type** *bytearray*

```
class cmt.cmap.v1.entities.Sphere
    Bases: cmt.cmap.a_entity.AEntity

    classmethod decode (data, offset, debug=False)
```

**Parameters**

- **data** (*bytes*) –
- **offset** (*int*) – without entity type byte
- **debug** (*bool*) –

```
encode ()
    Includes the entity type.
```

**Return type** *bytearray*

**cmt.cmap.v1.checkpoint\_time**

```
class cmt.cmap.v1.checkpoint_time.CheckpointTime (platin=0, gold=0, silver=0,  

    bronze=0)
```

Bases: *object*

```
cmt.cmap.v1.checkpoint_time.decode_checkpoint_times (data, offset, debug=False)
    Must start with the length byte.
```

**Parameters**

- **data** (bytes) –
- **offset** (int) –
- **debug** (bool) –

**Return type** `List[CheckpointTime]`

**cmt.converter**

**cmt.converter.v0**

**class** `cmt.converter.v0.Converter`

Bases: `cmt.a_converter.AConverter`

**static convert\_to** (*source, target*)

Convert to the other map format of same version.

**Return type** `Union[CMap, EMap]`

**static downgrade** (*source*)

Downgrade to the format version below.

**Return type** `None`

**static upgrade** (*source*)

Upgrade to the format version above.

**Return type** `Union[CMap, EMap]`

**cmt.converter.v1**

**class** `cmt.converter.v1.Converter`

Bases: `cmt.a_converter.AConverter`

**static convert\_to** (*source, target*)

Convert to the other map format of same version.

**Return type** `Union[CMap, EMap]`

**static downgrade** (*source*)

Downgrade to the format version below.

**Return type** `Union[CMap, EMap]`

**static upgrade** (*source*)

Upgrade to the format version above.

**Return type** `Union[Forwardref, Forwardref]`

**cmt.cs**

## cmt.cs.main

`cmt.cs.main.main` (*argv=None*)

Entry point into the program. Gets the arguments from the console and proceed them with `ArgumentParser`. Returns if its success successful 0 else 1.

## cmt.ecmap

### cmt.ecmap.a\_ecmap

**class** `cmt.ecmap.a_ecmap.AEMap` (*version*)

Bases: `cmt.a_map.AMap`

**abstract classmethod** `decode` (*data, offset, debug=False*)

Return type `AECMap`

**abstract** `encode` ()

Return type `bytearray`

### cmt.ecmap.v0

#### cmt.ecmap.v0.ecmap

**class** `cmt.ecmap.v0.ecmap.ECMap`

Bases: `cmt.ecmap.a_ecmap.AEMap`

Celaria .ecmap format (version 0)

#### Datatypes

Abbreviation	Type	Byte size
uByte	unsigned byte	1
uShort	unsigned short	2
uInt	unsigned int	4
sShort	signed short	2
sInt	signed int	4
f32	float	4
f64	double	8

#### Description format

> <datatype> (<number of datatypes in sequence>) // <description>

or

> [<variable name>] : <datatype> (<number of datatypes in sequence>) // <description>

Difference regarding to the .cmap begins with a '!!'.

#### Format

```

> uByte (11) // string identifier
> uByte (1) // version

> nameLen : uByte (1) // number of characters in map name
> uByte (nameLen) // map name as String

!! // checkpoint times are missing

> f32 (1) // sun rotation on Z axis
> f32 (1) // sun angle to xy plane (between 0 and 90 degrees)

... same as cmap v0 ...

```

**classmethod** `decode` (*data*, *offset*, *debug=False*)

**Return type** `ECMAP`

**encode** ()

**Return type** `bytearray`

### **cmt.ecmap.v1**

#### **cmt.ecmap.v1.ecmap**

**class** `cmt.ecmap.v1.ecmap.ECMap`

Bases: `cmt.ecmap.a_ecmap.AECMap`

Celaria .ecmap format (version 1)

#### **Datatypes**

Abbreviation	Type	Byte size
uByte	unsigned byte	1
uShort	unsigned short	2
uInt	unsigned int	4
sShort	signed short	2
sInt	signed int	4
f32	float	4
f64	double	8

#### **Description format**

```
> <datatype> (<number of datatypes in sequence>) // <description>
```

or

```
> [<variable name>] : <datatype> (<number of datatypes in sequence>) //
<description>
```

Difference regarding to the .cmap begins with a '!!'.

#### **Format**

```

> uByte (11) // string identifier
> uByte (1) // version

```

(continues on next page)



(continued from previous page)

```
> nameLen : uByte (1) // number of characters in map name
> uByte (nameLen) // map name as String

!! // checkpoint times are missing

> f32 (1) // sun rotation on Z axis
> f32 (1) // sun angle to xy plane (between 0 and 90 degrees)

... same as cmap v1 ...
```

**classmethod decode** (*data, offset, debug=False*)

**Return type** ECMAP

**encode** ()

**Return type** bytearray

- genindex
- modindex



## PYTHON MODULE INDEX

### C

- cmt.a\_converter, 4
- cmt.a\_map, 4
- cmt.blender.v2\_80.import\_menu, 6
- cmt.blender.v2\_80.object\_panel, 6
- cmt.blender.v2\_80.scene\_panel, 7
- cmt.blender.v2\_80.utils, 7
- cmt.blender.v2\_80.v1.add\_objects, 9
- cmt.blender.v2\_80.v1.codec, 10
- cmt.blender.v2\_80.viewport\_add\_menu, 8
- cmt.cmap.a\_cmap, 10
- cmt.cmap.a\_entity, 10
- cmt.cmap.v0.cmap, 10
- cmt.cmap.v0.entities, 12
- cmt.cmap.v0.medal\_time, 14
- cmt.cmap.v1.checkpoint\_time, 17
- cmt.cmap.v1.cmap, 14
- cmt.cmap.v1.entities, 16
- cmt.convert, 5
- cmt.converter.v0, 18
- cmt.converter.v1, 18
- cmt.cs.main, 19
- cmt.decode, 5
- cmt.ecmap.a\_ecmap, 19
- cmt.ecmap.v0.ecmap, 19
- cmt.ecmap.v1.ecmap, 20
- cmt.encode, 5
- cmt.static\_data, 5
- cmt.utils, 6



# INDEX

## Symbols

-h, -help  
command line option, 4  
-v, -version  
command line option, 4

## A

ACMap (class in *cmt.cmap.a\_cmap*), 10  
AConverter (class in *cmt.a\_converter*), 4  
add\_block() (in *module cmt.blender.v2\_80.v1.add\_objects*), 9  
add\_camera() (in *module cmt.blender.v2\_80.v1.add\_objects*), 9  
add\_dummy() (in *module cmt.blender.v2\_80.v1.add\_objects*), 9  
add\_exclusive\_to\_collection() (in *module cmt.blender.v2\_80.utils*), 7  
add\_player\_start() (in *module cmt.blender.v2\_80.v1.add\_objects*), 9  
add\_sphere() (in *module cmt.blender.v2\_80.v1.add\_objects*), 9  
add\_sun() (in *module cmt.blender.v2\_80.v1.add\_objects*), 9  
AddNewBlock (class in *module cmt.blender.v2\_80.viewport\_add\_menu*), 8  
AddNewDummy (class in *module cmt.blender.v2\_80.viewport\_add\_menu*), 8  
AddNewPlayerStart (class in *module cmt.blender.v2\_80.viewport\_add\_menu*), 8  
AddNewSphere (class in *module cmt.blender.v2\_80.viewport\_add\_menu*), 9  
AECMap (class in *cmt.ecmap.a\_ecmap*), 19  
AEntity (class in *cmt.cmap.a\_entity*), 10  
AMap (class in *cmt.a\_map*), 4  
AUTHOR (in *module cmt.static\_data*), 5  
AUTHOR\_EMAIL (in *module cmt.static\_data*), 5

## B

bl\_context (*cmt.blender.v2\_80.object\_panel.ObjectPanel* attribute), 7  
bl\_context (*cmt.blender.v2\_80.scene\_panel.ScenePanel* attribute), 7  
bl\_idname (*cmt.blender.v2\_80.import\_menu.ImportCMap* attribute), 6  
bl\_idname (*cmt.blender.v2\_80.object\_panel.ObjectPanel* attribute), 7  
bl\_idname (*cmt.blender.v2\_80.scene\_panel.ScenePanel* attribute), 7  
bl\_idname (*cmt.blender.v2\_80.viewport\_add\_menu.AddNewBlock* attribute), 8  
bl\_idname (*cmt.blender.v2\_80.viewport\_add\_menu.AddNewDummy* attribute), 8  
bl\_idname (*cmt.blender.v2\_80.viewport\_add\_menu.AddNewPlayerStart* attribute), 8  
bl\_idname (*cmt.blender.v2\_80.viewport\_add\_menu.AddNewSphere* attribute), 9  
bl\_idname (*cmt.blender.v2\_80.viewport\_add\_menu.ViewportAddMenu* attribute), 9  
bl\_label (*cmt.blender.v2\_80.import\_menu.ImportCMap* attribute), 6  
bl\_label (*cmt.blender.v2\_80.object\_panel.ObjectPanel* attribute), 7  
bl\_label (*cmt.blender.v2\_80.scene\_panel.ScenePanel* attribute), 7  
bl\_label (*cmt.blender.v2\_80.viewport\_add\_menu.AddNewBlock* attribute), 8  
bl\_label (*cmt.blender.v2\_80.viewport\_add\_menu.AddNewDummy* attribute), 8  
bl\_label (*cmt.blender.v2\_80.viewport\_add\_menu.AddNewPlayerStart* attribute), 8  
bl\_label (*cmt.blender.v2\_80.viewport\_add\_menu.AddNewSphere* attribute), 9  
bl\_label (*cmt.blender.v2\_80.viewport\_add\_menu.ViewportAddMenu* attribute), 9  
bl\_region\_type (*cmt.blender.v2\_80.object\_panel.ObjectPanel* attribute), 7  
bl\_region\_type (*cmt.blender.v2\_80.scene\_panel.ScenePanel* attribute), 7  
bl\_space\_type (*cmt.blender.v2\_80.object\_panel.ObjectPanel*

*attribute*), 7  
 bl\_space\_type (*cmt.blender.v2\_80.scene\_panel.ScenePanel attribute*), 7  
 Block (*class in cmt.cmap.v0.entities*), 12  
 Block (*class in cmt.cmap.v1.entities*), 16  
 BlockProps (*class cmt.blender.v2\_80.object\_panel*), 6  
 BlockType (*class in cmt.cmap.v0.entities*), 13  
 BlockType (*class in cmt.cmap.v1.entities*), 16

## C

CHECKPOINT (*cmt.cmap.v0.entities.BlockType attribute*), 13  
 CHECKPOINT (*cmt.cmap.v1.entities.BlockType attribute*), 16  
 CheckpointTime (*class cmt.cmap.v1.checkpoint\_time*), 17  
 CMap (*class in cmt.cmap.v0.cmap*), 10  
 CMap (*class in cmt.cmap.v1.cmap*), 14  
 CMAP (*cmt.a\_map.MapType attribute*), 5  
 cmt.a\_converter (*module*), 4  
 cmt.a\_map (*module*), 4  
 cmt.blender.v2\_80.import\_menu (*module*), 6  
 cmt.blender.v2\_80.object\_panel (*module*), 6  
 cmt.blender.v2\_80.scene\_panel (*module*), 7  
 cmt.blender.v2\_80.utils (*module*), 7  
 cmt.blender.v2\_80.v1.add\_objects (*module*), 9  
 cmt.blender.v2\_80.v1.codec (*module*), 10  
 cmt.blender.v2\_80.viewport\_add\_menu (*module*), 8  
 cmt.cmap.a\_cmap (*module*), 10  
 cmt.cmap.a\_entity (*module*), 10  
 cmt.cmap.v0.cmap (*module*), 10  
 cmt.cmap.v0.entities (*module*), 12  
 cmt.cmap.v0.medal\_time (*module*), 14  
 cmt.cmap.v1.checkpoint\_time (*module*), 17  
 cmt.cmap.v1.cmap (*module*), 14  
 cmt.cmap.v1.entities (*module*), 16  
 cmt.convert (*module*), 5  
 cmt.converter.v0 (*module*), 18  
 cmt.converter.v1 (*module*), 18  
 cmt.cs.main (*module*), 19  
 cmt.decode (*module*), 5  
 cmt.ecmap.a\_ecmap (*module*), 19  
 cmt.ecmap.v0.ecmap (*module*), 19  
 cmt.ecmap.v1.ecmap (*module*), 20  
 cmt.encode (*module*), 5  
 cmt.static\_data (*module*), 5  
 cmt.utils (*module*), 6  
 command line option  
   -h, -help, 4  
   -v, -version, 4  
   convert file {cmap,ecmap} {0,1} output, 4  
   convert file {cmap,ecmap} {0,1} output command line option, 4  
   convert () (*in module cmt.convert*), 5  
   convert\_to () (*cmt.a\_converter.AConverter static method*), 4  
   convert\_to () (*cmt.converter.v0.Converter static method*), 18  
   convert\_to () (*cmt.converter.v1.Converter static method*), 18  
   Converter (*class in cmt.converter.v0*), 18  
   Converter (*class in cmt.converter.v1*), 18  
   create\_material () (*in module cmt.blender.v2\_80.utils*), 7

## D

debug\_print () (*in module cmt.utils*), 6  
 DebugIterUnpack (*class in cmt.utils*), 6  
 decode () (*cmt.a\_map.AMap class method*), 4  
 decode () (*cmt.cmap.a\_cmap.ACMap class method*), 10  
 decode () (*cmt.cmap.a\_entity.AEntity class method*), 10  
 decode () (*cmt.cmap.v0.cmap.CMap class method*), 12  
 decode () (*cmt.cmap.v0.entities.Block class method*), 12  
 decode () (*cmt.cmap.v0.entities.Dummy class method*), 13  
 decode () (*cmt.cmap.v0.entities.PlayerStart class method*), 13  
 decode () (*cmt.cmap.v0.entities.Sphere class method*), 13  
 decode () (*cmt.cmap.v1.cmap.CMap class method*), 16  
 decode () (*cmt.cmap.v1.entities.Block class method*), 16  
 decode () (*cmt.cmap.v1.entities.Dummy class method*), 17  
 decode () (*cmt.cmap.v1.entities.PlayerStart class method*), 17  
 decode () (*cmt.cmap.v1.entities.Sphere class method*), 17  
 decode () (*cmt.ecmap.a\_ecmap.AECMap class method*), 19  
 decode () (*cmt.ecmap.v0.ecmap.ECMap class method*), 20  
 decode () (*cmt.ecmap.v1.ecmap.ECMap class method*), 21  
 decode () (*in module cmt.blender.v2\_80.v1.codec*), 10  
 decode () (*in module cmt.decode*), 5  
 decode\_checkpoint\_times () (*in module cmt.cmap.v1.checkpoint\_time*), 17  
 decode\_medal\_times () (*in module cmt.cmap.v0.medal\_time*), 14  
 DESCRIPTION (*in module cmt.static\_data*), 5

downgrade() (*cmt.a\_converter.AConverter* static method), 4  
 downgrade() (*cmt.converter.v0.Converter* static method), 18  
 downgrade() (*cmt.converter.v1.Converter* static method), 18  
 draw() (*cmt.blender.v2\_80.object\_panel.ObjectPanel* method), 7  
 draw() (*cmt.blender.v2\_80.scene\_panel.ScenePanel* method), 7  
 draw() (*cmt.blender.v2\_80.viewport\_add\_menu.ViewportAddMenu* method), 9  
 draw\_header() (*cmt.blender.v2\_80.object\_panel.ObjectPanel* method), 7  
 draw\_header() (*cmt.blender.v2\_80.scene\_panel.ScenePanel* method), 7  
 Dummy (class in *cmt.blender.v2\_80.object\_panel*), 6  
 Dummy (class in *cmt.cmap.v0.entities*), 13  
 Dummy (class in *cmt.cmap.v1.entities*), 16

## E

EMap (class in *cmt.ecmap.v0.ecmap*), 19  
 EMap (class in *cmt.ecmap.v1.ecmap*), 20  
 EMap (*cmt.a\_map.MapType* attribute), 5  
 encode() (*cmt.a\_map.AMap* method), 4  
 encode() (*cmt.cmap.a\_cmap.AMap* method), 10  
 encode() (*cmt.cmap.a\_entity.AEntity* method), 10  
 encode() (*cmt.cmap.v0.cmap.CMap* method), 12  
 encode() (*cmt.cmap.v0.entities.Block* method), 12  
 encode() (*cmt.cmap.v0.entities.Dummy* method), 13  
 encode() (*cmt.cmap.v0.entities.PlayerStart* method), 13  
 encode() (*cmt.cmap.v0.entities.Sphere* method), 14  
 encode() (*cmt.cmap.v1.cmap.CMap* method), 16  
 encode() (*cmt.cmap.v1.entities.Block* method), 16  
 encode() (*cmt.cmap.v1.entities.Dummy* method), 17  
 encode() (*cmt.cmap.v1.entities.PlayerStart* method), 17  
 encode() (*cmt.cmap.v1.entities.Sphere* method), 17  
 encode() (*cmt.ecmap.a\_ecmap.AEMap* method), 19  
 encode() (*cmt.ecmap.v0.ecmap.EMap* method), 20  
 encode() (*cmt.ecmap.v1.ecmap.EMap* method), 21  
 encode() (in module *cmt.encode*), 5  
 execute() (*cmt.blender.v2\_80.import\_menu.ImportCMap* method), 6  
 execute() (*cmt.blender.v2\_80.viewport\_add\_menu.AddNewBlock* method), 8  
 execute() (*cmt.blender.v2\_80.viewport\_add\_menu.AddNewDummy* method), 8  
 execute() (*cmt.blender.v2\_80.viewport\_add\_menu.AddNewPlayerStart* method), 8  
 execute() (*cmt.blender.v2\_80.viewport\_add\_menu.AddNewSphere* method), 9

## F

filename\_ext (*cmt.blender.v2\_80.import\_menu.ImportCMap* attribute), 6  
 FINISH (*cmt.cmap.v0.entities.BlockType* attribute), 13  
 FINISH (*cmt.cmap.v1.entities.BlockType* attribute), 16  
 from\_str (*cmt.a\_map.MapType* attribute), 5

## G

get\_collection() (in module *cmt.blender.v2\_80.utils*), 7  
 get\_color() (in module *cmt.blender.v2\_80.v1.add\_objects*), 9

## I

ICE (*cmt.cmap.v0.entities.BlockType* attribute), 13  
 ICE (*cmt.cmap.v1.entities.BlockType* attribute), 16  
 import\_cmap() (in module *cmt.blender.v2\_80.import\_menu*), 6  
 ImportCMap (class in *cmt.blender.v2\_80.import\_menu*), 6

## J

JUMP (*cmt.cmap.v0.entities.BlockType* attribute), 13  
 JUMP (*cmt.cmap.v1.entities.BlockType* attribute), 16

## L

LONG\_NAME (in module *cmt.static\_data*), 5

## M

main() (in module *cmt.cs.main*), 19  
 MapType (class in *cmt.a\_map*), 4  
 MedalTime (class in *cmt.cmap.v0.medal\_time*), 14  
 menu\_add\_object() (in module *cmt.blender.v2\_80.viewport\_add\_menu*), 9  
 menu\_func\_import() (in module *cmt.blender.v2\_80.import\_menu*), 6  
 MetadataProps (class in *cmt.blender.v2\_80.scene\_panel*), 7

## N

NAME (in module *cmt.static\_data*), 5  
 NOTHING (*cmt.cmap.v0.entities.BlockType* attribute), 13  
 NOTHING (*cmt.cmap.v1.entities.BlockType* attribute), 16

## O

ObjectPanel (class in *cmt.blender.v2\_80.object\_panel*), 7

## P

PlayerStart (class in *cmt.cmap.v0.entities*), 13  
 PlayerStart (class in *cmt.cmap.v1.entities*), 17

poll() (*cmt.blender.v2\_80.object\_panel.ObjectPanel class method*), 7  
 PROJECT\_URL (*in module cmt.static\_data*), 5

## R

register() (*in module cmt.blender.v2\_80.import\_menu*), 6  
 register() (*in module cmt.blender.v2\_80.object\_panel*), 7  
 register() (*in module cmt.blender.v2\_80.scene\_panel*), 7  
 register() (*in module cmt.blender.v2\_80.viewport\_add\_menu*), 9

## S

ScenePanel (*class in cmt.blender.v2\_80.scene\_panel*), 7  
 SPEED (*cmt.cmap.v0.entities.BlockType attribute*), 13  
 SPEED (*cmt.cmap.v1.entities.BlockType attribute*), 16  
 Sphere (*class in cmt.cmap.v0.entities*), 13  
 Sphere (*class in cmt.cmap.v1.entities*), 17

## T

to\_bl\_location() (*in module cmt.blender.v2\_80.utils*), 8  
 to\_bl\_rotation\_z() (*in module cmt.blender.v2\_80.utils*), 8  
 to\_bl\_scale() (*in module cmt.blender.v2\_80.utils*), 8  
 to\_cmt\_position() (*in module cmt.blender.v2\_80.utils*), 8  
 to\_cmt\_rotation\_z() (*in module cmt.blender.v2\_80.utils*), 8  
 to\_cmt\_scale() (*in module cmt.blender.v2\_80.utils*), 8  
 to\_hex() (*in module cmt.utils*), 6

## U

unpack\_from() (*in module cmt.utils*), 6  
 unregister() (*in module cmt.blender.v2\_80.import\_menu*), 6  
 unregister() (*in module cmt.blender.v2\_80.object\_panel*), 7  
 unregister() (*in module cmt.blender.v2\_80.scene\_panel*), 7  
 unregister() (*in module cmt.blender.v2\_80.viewport\_add\_menu*), 9  
 upgrade() (*cmt.a\_converter.AConverter static method*), 4  
 upgrade() (*cmt.converter.v0.Converter static method*), 18

## V

VERSION (*in module cmt.static\_data*), 5  
 ViewportAddMenu (*class in cmt.blender.v2\_80.viewport\_add\_menu*), 9