
$WX_{i\text{cons}_h\text{icolor}}$
Release 0.2.0

Dominic Davis-Foster

Nov 27, 2020

DOCUMENTATION

1	Installation	3
1.1	Usage	3
1.2	API Reference	3
1.2.1	<code>wx_icons_hicolor.constants</code>	4
1.2.2	<code>wx_icons_hicolor.directory</code>	5
1.2.3	<code>wx_icons_hicolor.icon</code>	6
1.2.4	<code>wx_icons_hicolor.icon_theme</code>	7
1.3	Overview	9
1.4	Coding style	10
1.5	Automated tests	10
1.6	Type Annotations	10
1.7	Build documentation locally	10
1.8	Downloading source code	10
1.8.1	Building from source	11
	Python Module Index	13
	Index	15

Hicolor icon theme for wxPython.

Docs	
Tests	
PyPI	
Activity	
Other	

INSTALLATION

from PyPI

```
$ python3 -m pip install wx_icons_hicolor --user
```

from GitHub

```
$ python3 -m pip install git+https://github.com/domdfcoding/custom_wx_icons_hicolor@master --user
```

1.1 Usage

To use `wx_icons_hicolor` in your application:

```
from wx_icons_hicolor import wxHicolorIconTheme

class MyApp(wx.App):
    def OnInit(self):
        wx.ArtProvider.Push(wxHicolorIconTheme())
        self.frame = TestFrame(None, wx.ID_ANY)
        self.SetTopWindow(self.frame)
        self.frame.Show()
        return True
```

And then the icons can be accessed through `wx.ArtProvider`:

```
wx.ArtProvider.GetBitmap('document-new', wx.ART_OTHER, wx.Size(48, 48))
```

Any `FreeDesktop Icon Theme Specification` name can be used.

Currently the *Client ID* is not used, so just pass `wx.ART_OTHER`.

1.2 API Reference

version()

Returns the library and theme versions.

Return type `str`

class `wxHicolorIconTheme`

Bases: `ArtProvider`

`wx.ArtProvider` subclass providing the Hicolor Icon Theme.

Methods:

<code>CreateBitmap(id, client, size)</code>	Returns the requested resource.
<code>icon2bitmap(icon, size)</code>	Converts an <i>Icon</i> to a <code>wx.Bitmap</code> .

CreateBitmap (*id*, *client*, *size*)

Returns the requested resource.

Parameters

- **id** (*Any*) – Unique identifier of the bitmap.
- **client** (*Any*) – Identifier of the client (i.e. who is asking for the bitmap). This only serves as a hint.
- **size** (`Union[Tuple[int], Size]`) – Preferred size of the bitmap. The function may return a bitmap of different dimensions; it will be automatically rescaled to meet client's request.

Return type `Bitmap`

static icon2bitmap (*icon*, *size*)

Converts an *Icon* to a `wx.Bitmap`.

Parameters

- **icon** (*Icon*)
- **size** (`int`) – The desired size of the icon. If the icon isn't scalable the icon is returned in its original size.

Return type `Bitmap`

1.2.1 wx_icons_hicolor.constants

Constants for use in `wx_icons_hicolor` and its derivatives.

Data:

<code>IconTypes</code>	Type hint for valid icon type strings.
<code>mime</code>	Instance of <code>magic.Magic</code> to identify mimetypes of files.
<code>theme_index_path</code>	Path to the theme index file.

IconTypes

Type hint for valid icon type strings.

Alias of `Literal['Fixed', 'Scalable', 'Threshold']`

mime

Type: `Magic`

Instance of `magic.Magic` to identify mimetypes of files.

theme_index_path

Type: `PosixPath`

Path to the theme index file.

1.2.2 wx_icons_hicolor.directory

Class to represent directory of icons.

Classes:

<code>Directory</code> (path, size[, scale, context, ...])	Represents a directory containing icons.
--	--

class `Directory` (path, size, scale=1, context="", type='Threshold', max_size=None, min_size=None, threshold=2, theme="")

Bases: `Dictable`

Represents a directory containing icons.

Parameters

- **path** (`Union[str, Path, PathLike]`) – The absolute path to the directory
- **size** (`int`) – Nominal (unscaled) size of the icons in this directory.
- **scale** (`int`) – Target scale of the icons in this directory. Any directory with a scale other than 1 should be listed in the `ScaledDirectories` list rather than `Directories` for backwards compatibility. Default 1.
- **context** (`str`) – The context the icon is normally used in. This is in detail discussed in the section called “Context”. Default ' '.
- **type** (`Literal['Fixed', 'Scalable', 'Threshold']`) – The type of icon sizes for the icons in this directory. Valid types are 'Fixed', 'Scalable' and 'Threshold'. The type decides what other keys in the section are used. Default 'Threshold'.
- **max_size** (`Optional[int]`) – Specifies the maximum (unscaled) size that the icons in this directory can be scaled to. Defaults to the value of `Size` if not present.
- **min_size** (`Optional[int]`) – Specifies the minimum (unscaled) size that the icons in this directory can be scaled to. Defaults to the value of `Size` if not present.
- **threshold** (`int`) – The icons in this directory can be used if the size differ at most this much from the desired (unscaled) size. Defaults to 2 if not present. Default 2.
- **theme** (`str`) – The name of the theme this directory is a part of. Default ' '.

Methods:

<code>__repr__</code> ()	Return a string representation of the <code>Directory</code> .
<code>from_configparser</code> (config_section, ...)	Constructs a <code>Directory</code> from a <code>configparser</code> section.

Attributes:

<code>icons</code>	Returns a list of icons in this <code>Directory</code> .
--------------------	--

`__repr__`()
Return a string representation of the `Directory`.

Return type `str`

classmethod `from_configparser` (config_section, theme_content_root)
Constructs a `Directory` from a `configparser` section.

Parameters

- **config_section** (SectionProxy)
- **theme_content_root** (Path)

Return type *Directory*

property icons

Returns a list of icons in this *Directory*.

Return type *List[Icon]*

1.2.3 wx_icons_hicolor.icon

Class to represent icons.

Classes:

<i>Icon</i> (name, path, size[, type, max_size, ...])	Represents an icon.
---	---------------------

class *Icon* (name, path, size, type='Threshold', max_size=None, min_size=None, theme='')

Bases: *Dictable*

Represents an icon.

Parameters

- **name** (*str*) – The name of the icon.
- **path** (*Union[str, Path, PathLike]*) – The path to the icon.
- **size** (*int*) – Nominal (unscaled) size of the icon.
- **type** (*Literal['Fixed', 'Scalable', 'Threshold']*) – The type of icon sizes for the icon. Valid types are 'Fixed', 'Scalable' and 'Threshold'. The type decides what other keys in the section are used. Default 'Threshold'.
- **max_size** (*Optional[int]*) – Specifies the maximum (unscaled) size that the icon can be scaled to. Defaults to the value of *Size* if not present.
- **min_size** (*Optional[int]*) – Specifies the minimum (unscaled) size that the icon can be scaled to. Defaults to the value of *Size* if not present.
- **theme** (*str*) – The name of the theme this icon came from. Default ' '.

Methods:

<i>__eq__</i> (other)	Return self == other.
<i>__repr__</i> ()	Return a string representation of the <i>Icon</i> .
<i>as_base64_png</i> ([size])	Returns the icon as a base64-encoded object containing PNG image data.
<i>as_bitmap</i> ([size])	Returns the icon as a wxPython bitmap.
<i>as_png</i> ([size])	Returns the icon as a <i>BytesIO</i> object containing PNG image data.

Attributes:

<code>mime_type</code>	The mime type of the icon.
<code>scalable</code>	Whether the icon is scalable.

`__eq__(other)`

Return `self == other`.

Return type `bool`

`__repr__()`

Return a string representation of the *Icon*.

Return type `str`

`as_base64_png(size=None)`

Returns the icon as a base64-encoded object containing PNG image data.

Return type `str`

Returns Base64-encoded string representing the PNG image.

`as_bitmap(size=None)`

Returns the icon as a wxPython bitmap.

Parameters `size` (`Optional[int]`) – Default `None`.

Return type `Bitmap`

`as_png(size=None)`

Returns the icon as a `BytesIO` object containing PNG image data.

Return type `BytesIO`

Returns `io.BytesIO` object representing the PNG image.

property `mime_type`

The mime type of the icon.

Return type `str`

property `scalable`

Whether the icon is scalable.

Return type `bool`

1.2.4 wx_icons_{hicolor}.icon_theme

Class to represent an icon theme.

Classes:

<code>HicolorIconTheme(name, comment, directories)</code>	The Hicolor Icon Theme.
<code>IconTheme(name, comment, directories[, ...])</code>	Represents an icon theme.

class `HicolorIconTheme` (*name, comment, directories, inherits=None, scaled_directories=None, hidden=False, example=""*)

Bases: `IconTheme`

The Hicolor Icon Theme.

Parameters

- **name** (`str`) – short name of the icon theme, used in e.g. lists when selecting themes.

- **comment** (*str*) – longer string describing the theme
- **inherits** (*Optional[Sequence[str]]*) – The name of the theme that this theme inherits from. If an icon name is not found in the current theme, it is searched for in the inherited theme (and recursively in all the inherited themes). Default *None*.

If no theme is specified implementations are required to add the “hicolor” theme to the inheritance tree. An implementation may optionally add other default themes in between the last specified theme and the hicolor theme.

- **directories** (*Sequence[Directory]*) – list of subdirectories for this theme. For every subdirectory there must be a section in the index.theme file describing that directory.
- **scaled_directories** (*Optional[Sequence[Directory]]*) – Additional list of subdirectories for this theme, in addition to the ones in Directories. These directories should only be read by implementations supporting scaled directories and was added to keep compatibility with old implementations that don’t support these. Default *None*.
- **hidden** (*bool*) – Whether to hide the theme in a theme selection user interface. This is used for things such as fallback-themes that are not supposed to be visible to the user. Default *False*.
- **example** (*str*) – The name of an icon that should be used as an example of how this theme looks. Default ' '.

Methods:

<i>create()</i>	Create an instance of the Hicolor Icon Theme.
-----------------	---

classmethod *create()*

Create an instance of the Hicolor Icon Theme.

Return type *HicolorIconTheme*

class *IconTheme* (*name, comment, directories, inherits=None, scaled_directories=None, hidden=False, example=""*)

Bases: *Dictable*

Represents an icon theme.

Parameters

- **name** (*str*) – short name of the icon theme, used in e.g. lists when selecting themes.
- **comment** (*str*) – longer string describing the theme
- **inherits** (*Optional[Sequence[str]]*) – The name of the theme that this theme inherits from. If an icon name is not found in the current theme, it is searched for in the inherited theme (and recursively in all the inherited themes). Default *None*.

If no theme is specified implementations are required to add the “hicolor” theme to the inheritance tree. An implementation may optionally add other default themes in between the last specified theme and the hicolor theme.

- **directories** (*Sequence[Directory]*) – list of subdirectories for this theme. For every subdirectory there must be a section in the index.theme file describing that directory.
- **scaled_directories** (*Optional[Sequence[Directory]]*) – Additional list of subdirectories for this theme, in addition to the ones in Directories. These directories should only be read by implementations supporting scaled directories and was added to keep compatibility with old implementations that don’t support these. Default *None*.

- **hidden** (`bool`) – Whether to hide the theme in a theme selection user interface. This is used for things such as fallback-themes that are not supposed to be visible to the user. Default `False`.
- **example** (`str`) – The name of an icon that should be used as an example of how this theme looks. Default `''`.

Methods:

<code>__repr__()</code>	Return a string representation of the <i>IconTheme</i> .
<code>__str__()</code>	Return <code>str(self)</code> .
<code>find_icon(icon_name, size, scale[, ...])</code>	Searches for the icon with the given name and size.
<code>from_configparser(theme_index_path)</code>	Constructs a <i>IconTheme</i> from config file.

`__repr__()`
Return a string representation of the *IconTheme*.

Return type `str`

`__str__()`
Return `str(self)`.

Return type `str`

`find_icon(icon_name, size, scale, prefer_this_theme=True)`
Searches for the icon with the given name and size.

Parameters

- **icon_name** (`str`) – The name of the icon to find. Any [FreeDesktop Icon Theme Specification](#) name can be used.
- **size** (`int`) – The desired size of the icon
- **scale** (`Any`) – TODO: Currently does nothing
- **prefer_this_theme** (`bool`) – Return an icon from this theme even if it has to be resized, rather than a correctly sized icon from the parent theme. Default `True`.

Return type `Optional[Icon]`

Returns The icon if it was found, or `None`.

classmethod `from_configparser(theme_index_path)`
Constructs a *IconTheme* from config file.

Parameters `theme_index_path` (`Union[str, Path, PathLike]`)

Return type *IconTheme*

1.3 Overview

`wx_icons_hicolor` uses `tox` to automate testing and packaging, and `pre-commit` to maintain code quality.

Install `pre-commit` with `pip` and install the git hook:

```
$ python -m pip install pre-commit
$ pre-commit install
```

1.4 Coding style

yapf-isort is used for code formatting.

It can be run manually via pre-commit:

```
$ pre-commit run yapf-isort -a
```

Or, to run the complete autoformatting suite:

```
$ pre-commit run -a
```

1.5 Automated tests

Tests are run with tox and pytest. To run tests for a specific Python version, such as Python 3.6, run:

```
$ tox -e py36
```

To run tests for all Python versions, simply run:

```
$ tox
```

1.6 Type Annotations

Type annotations are checked using mypy. Run mypy using tox:

```
$ tox -e mypy
```

1.7 Build documentation locally

The documentation is powered by Sphinx. A local copy of the documentation can be built with tox:

```
$ tox -e docs
```

1.8 Downloading source code

The wx_icons_hicolor source code is available on GitHub, and can be accessed from the following URL: https://github.com/domdfcoding/custom_wx_icons_hicolor

If you have git installed, you can clone the repository with the following command:

```
$ git clone https://github.com/domdfcoding/custom_wx_icons_hicolor"
> Cloning into 'custom_wx_icons_hicolor'...
> remote: Enumerating objects: 47, done.
> remote: Counting objects: 100% (47/47), done.
> remote: Compressing objects: 100% (41/41), done.
> remote: Total 173 (delta 16), reused 17 (delta 6), pack-reused 126
> Receiving objects: 100% (173/173), 126.56 KiB | 678.00 KiB/s, done.
> Resolving deltas: 100% (66/66), done.
```

Alternatively, the code can be downloaded in a ‘zip’ file by clicking:

Clone or download → Download Zip

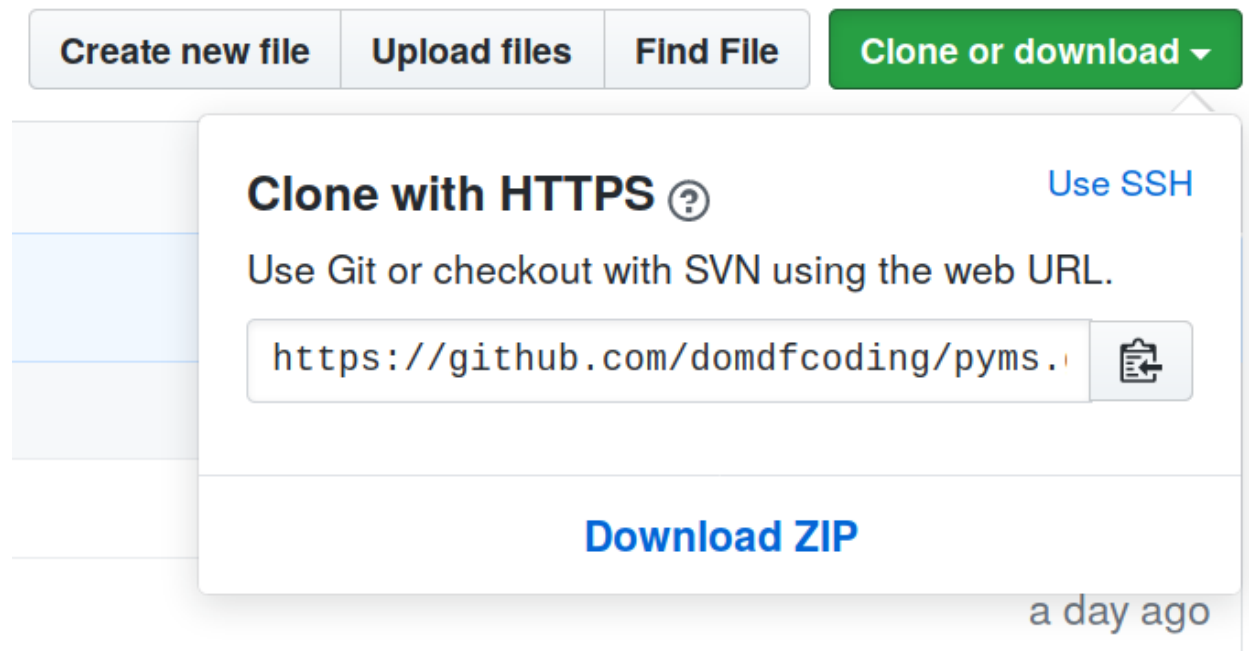


Fig. 1: Downloading a ‘zip’ file of the source code

1.8.1 Building from source

The recommended way to build wx_icons_hicolor is to use `tox`:

```
$ tox -e build
```

The source and wheel distributions will be in the directory `dist`.

If you wish, you may also use `pep517.build` or another **PEP 517**-compatible build tool.

View the [Function Index](#) or browse the [Source Code](#).

[Browse the GitHub Repository](#)

PYTHON MODULE INDEX

W

`wx_icons_hicolor.__init__`, 3
`wx_icons_hicolor.constants`, 4
`wx_icons_hicolor.directory`, 5
`wx_icons_hicolor.icon`, 6
`wx_icons_hicolor.icon_theme`, 7

Symbols

`__eq__()` (*Icon method*), 7
`__repr__()` (*Directory method*), 5
`__repr__()` (*Icon method*), 7
`__repr__()` (*IconTheme method*), 9
`__str__()` (*IconTheme method*), 9

A

`as_base64_png()` (*Icon method*), 7
`as_bitmap()` (*Icon method*), 7
`as_png()` (*Icon method*), 7

C

`create()` (*HicolorIconTheme class method*), 8
`CreateBitmap()` (*wxHicolorIconTheme method*), 4

D

`Directory` (*class in wx_icons_hicolor.directory*), 5

F

`find_icon()` (*IconTheme method*), 9
`from_configparser()` (*Directory class method*), 5
`from_configparser()` (*IconTheme class method*), 9

H

`HicolorIconTheme` (*class in wx_icons_hicolor.icon_theme*), 7

I

`Icon` (*class in wx_icons_hicolor.icon*), 6
`icon2bitmap()` (*wxHicolorIconTheme static method*), 4
`icons()` (*Directory property*), 6
`IconTheme` (*class in wx_icons_hicolor.icon_theme*), 8
`IconTypes` (*in module wx_icons_hicolor.constants*), 4

M

`mime` (*in module wx_icons_hicolor.constants*), 4
`mime_type()` (*Icon property*), 7
`module`

`wx_icons_hicolor.__init__`, 3
`wx_icons_hicolor.constants`, 4
`wx_icons_hicolor.directory`, 5
`wx_icons_hicolor.icon`, 6
`wx_icons_hicolor.icon_theme`, 7

P

Python Enhancement Proposals
 PEP 517, 11

S

`scalable()` (*Icon property*), 7

T

`theme_index_path` (*in module wx_icons_hicolor.constants*), 4

V

`version()` (*in module wx_icons_hicolor.__init__*), 3

W

`wx_icons_hicolor.__init__`
module, 3
`wx_icons_hicolor.constants`
module, 4
`wx_icons_hicolor.directory`
module, 5
`wx_icons_hicolor.icon`
module, 6
`wx_icons_hicolor.icon_theme`
module, 7
`wxHicolorIconTheme` (*class in wx_icons_hicolor.__init__*), 3