

---

# **Piccolo Theme**

**Daniel Townsend**

**Mar 23, 2022**



**CONTENTS:**

<b>1</b>	<b>Setup</b>	<b>3</b>
<b>2</b>	<b>Configuration</b>	<b>5</b>
<b>3</b>	<b>Help</b>	<b>7</b>
<b>4</b>	<b>Support Us</b>	<b>9</b>
<b>5</b>	<b>About</b>	<b>11</b>
<b>6</b>	<b>Contributing</b>	<b>13</b>
<b>7</b>	<b>Demo</b>	<b>15</b>
<b>8</b>	<b>Changes</b>	<b>19</b>
	<b>Index</b>	<b>21</b>



A clean and modern theme for [Sphinx](#).



## 1.1 Install Sphinx

```
pip install sphinx
```

Create a docs folder within your project, and run `sphinx-quickstart`.

```
mkdir docs  
cd docs  
sphinx-quickstart
```

`sphinx-quickstart` will scaffold your documentation for you.

---

## 1.2 Install Piccolo Theme

```
pip install piccolo_theme
```

Find the `conf.py` file that Sphinx generated for you. Within that set the following value:

```
html_theme = 'piccolo_theme'
```

---

## 1.3 Building your docs

Sphinx creates a `Makefile` in your docs folder. To generate some HTML docs, run the following in the same directory as your `Makefile`:

```
make html
```

Within your docs folder, Sphinx will have generated some HTML files in `_build/html`.

To serve these files using Python, you can use:

```
python -m http.server --directory _build/html/
```

Now open up <http://localhost:8000> in your browser to see your beautiful docs!

---





## CONFIGURATION

### 2.1 html\_short\_title

If you have a really long project name, you may prefer something shorter to appear in the navigation bar. Specify this using `html_short_title` in `conf.py`:

```
# conf.py

# By default the project value is used in the nav bar.
project = 'My Extra Special Amazing Docs'

# If specified, this will be used in the nav bar instead.
html_short_title = "Amazing Docs"
```



The best place to get help is on our [GitHub discussions page](#).

It's also a great place to share any feedback you have about the theme, and how we can make improvements.



## SUPPORT US

You can help us in many ways:

- Sharing the project with others
- Leaving [feedback](#)
- Starring the repo on [GitHub](#)
- Making improvements to the theme by making a [pull request](#)

Thanks!



## ABOUT

This theme was created to document [Piccolo](#) and sister projects.

Here's a live example of it being used:

- <https://piccolo-orm.readthedocs.io/en/latest/>





## CONTRIBUTING

### 6.1 Styles

We use [Sass](#) as a CSS preprocessor. The styles live in `src/sass`.

To modify the styles, first install Sass:

```
npm install -g sass
```

Then run:

```
./scripts/build-styles.sh
```

This will automatically rebuild your CSS when it detects a change in the Sass files.

---

### 6.2 Running the docs

By running Piccolo Theme's docs you can verify that your changes look OK.

First install the requirements:

```
pip install -r requirements/doc-requirements.txt
```

Then launch a web server using the following script:

```
./scripts/run-docs.sh
```

It auto reloads when it detects changes to the documentation or theme files.



This is used to demonstrate various features, and also for testing.

---

## 7.1 Autodoc

```
class piccolo_theme.snippets.Column(null: bool = False, primary_key: bool = False, unique: bool = False,  
                                     index: bool = False, required: bool = False, help_text: Optional[str]  
                                     = None, choices: Optional[Type[enum.Enum]] = None,  
                                     db_column_name: Optional[str] = None, secret: bool = False,  
                                     **kwargs)
```

All other columns inherit from `Column`. Don't use it directly.

The following arguments apply to all column types:

### Parameters

- **null** – Whether the column is nullable.
- **primary\_key** – If set, the column is used as a primary key.
- **default** – The column value to use if not specified by the user.
- **unique** – If set, a unique constraint will be added to the column.
- **index** – Whether an index is created for the column, which can improve the speed of selects, but can slow down inserts.
- **index\_method** – If index is set to True, this specifies what type of index is created.
- **required** – This isn't used by the database - it's to indicate to other tools that the user must provide this value. Example uses are in serialisers for API endpoints, and form fields.
- **help\_text** – This provides some context about what the column is being used for. For example, for a `Decimal` column called `value`, it could say 'The units are millions of dollars'. The database doesn't use this value, but tools such as Piccolo Admin use it to show a tooltip in the GUI.
- **choices** – An optional Enum - when specified, other tools such as Piccolo Admin will render the available options in the GUI.
- **db\_column\_name** – If specified, you can override the name used for the column in the database. The main reason for this is when using a legacy database, with a problematic column name (for example 'class', which is a reserved Python keyword). Here's an example:

```
class MyTable(Table):
    class_ = Varchar(db_column_name="class")

>>> await MyTable.select(MyTable.class_)
[{'id': 1, 'class': 'test'}]
```

This is an advanced feature which you should only need in niche situations.

- **secret** – If `secret=True` is specified, it allows a user to automatically omit any fields when doing a select query, to help prevent inadvertent leakage of sensitive data.

```
class Band(Table):
    name = Varchar()
    net_worth = Integer(secret=True)

>>> await Band.select(exclude_secrets=True)
[{'name': 'Pythonistas'}]
```

---

## 7.2 Breathe

class **CppClass**

*CppClass* class.

Details about *CppClass*.

### Public Functions

const char \***member\_function**(char, int)

A member function.

#### Parameters

- **c** – a character.
- **n** – an integer.

**Throws** `std::out_of_range` – parameter is out of range.

**Returns** a character pointer.

void **cpp\_function**(int \*a, int \*b, int \*c)

Doing important things with parameter directions.

#### Parameters

- **a** – [out] output
- **b** – [in] input
- **c** – [inout] input but gets rewritten

## 7.3 Tables

### 7.3.1 Table 1

Name	Drives
Alice	True
Bob	True
Curtis	False

### 7.3.2 Table 2

Header 1	Header 2	Header 3
body row 1	column 2	column 3
body row 2	Cells may span columns. And several paragraphs.	
body row 3	Cells may span rows.	<ul style="list-style-type: none"><li>• Cells</li><li>• contain</li><li>• blocks.</li></ul>
body row 4		

---

## 7.4 Data definitions

**Python** A great programming language.

**Sphinx** A powerful documentation tool.

---

## 7.5 Lists

### 7.5.1 Unordered List

- Python
- Rust
- Javascript

### 7.5.2 Ordered List

1. Python
  2. Rust
  3. Javascript
- 

## 7.6 Admonitions

**Warning:** This is a warning!

**Error:** This is an error!

**Hint:** This is a hint!

**Note:** This is a note!

## CHANGES

### 8.1 0.6.0

If `html_short_title` is in `conf.py` then this is used in the nav bar instead of the full project title.

---

### 8.2 0.5.1

Fixed dark mode styles - some elements weren't visible. Thanks to @alorence for reporting this issue.

---

### 8.3 0.5.0

Added table styles.

---

### 8.4 0.4.0

Improved the appearance of autodoc output for C++ files (when using [breathe](#)). Courtesy @thijsmie.

---

### 8.5 0.3.0

Added dark mode.

---

## **8.6 0.2.5**

Improved search styles.

---

## **8.7 0.2.4**

Added missing `requirements.txt` file to manifest. Thanks to @moorepants for reporting this.

---

## **8.8 0.2.3**

Make the `page contents` text smaller when the right hand sidebar is hidden.

---

## **8.9 0.2.2**

Fix missing static files.

---

## **8.10 0.2.1**

Fix missing static files.

---

## **8.11 0.2.0**

Improved the main header on mobile - the search bar is replaced with a search icon. Also increased the size of the touch targets for showing / hiding the right sidebar, for easier use on mobile. See [PR 7](#).



## INDEX

### C

`Column` (*class in piccolo\_theme.snippets*), 15  
`cpp_function` (*C++ function*), 16  
`CppClass` (*C++ class*), 16  
`CppClass::member_function` (*C++ function*), 16