

# Augmented Docs

a love letter to **rustdoc** and **docs.rs**

# Who Am I?

- François Mockers
- QA Lead at PayLead
- Maintainer of Bevy
- Developer of random things
- Find me at [@FrancoisMockers@hachyderm.io](mailto:@FrancoisMockers@hachyderm.io)

# Documenting your Code

# Why?

- Enhances discoverability
  - Helps users find your crate
  - And check that it matches their needs
- Improves usability
  - Clarifies the purpose and functionality of your code
  - Reduces the learning curve for new users

# Documenting a Function

- What is it doing?
- Explain its parameters and return value
- Detail the error cases and the panics
- Unsafe? Give the conditions the caller has to check
- Example usage

# Documenting a Struct

- Describe each field and its purpose
- Specify default values
- Explain any constraints or invariants
- Mention any related structs or traits
- Example usage

# Documenting a Trait

- Explain the purpose and use cases of the trait
- Describe the expected behavior when implemented
- Detail the associated types
- Highlight relationships with other traits
- Discuss default method implementations
- Example implementation

# Documenting a Module or a Crate

- Provide an overview of the module or crate's functionality
- Explain how the different components interact and fit together
- Highlight key features and capabilities
- Include examples of common usage patterns
- Guide users on where to find specific functionalities or components



**rustdoc**

# The Tool

- You all know it!

```
rustdoc
```

```
cargo doc
```

- In rust-lang/rust/src/librustdoc
- The rustdoc book: <https://doc.rust-lang.org/rustdoc>

# Rustdoc team

Developing and managing the Rustdoc documentation tool

#T-RUSTDOC ON ZULIP

## Members



**Guillaume Gomez**  
GitHub: [GuillaumeGomez](#)  
Team leader



**Alona Enraght-Moony**  
GitHub: [aDotInTheVoid](#)



**Noah Lev**  
GitHub: [camelid](#)



**León Orell Valerian Liehr**  
GitHub: [fmease](#)



**Jacob Hoffman-Andrews**  
GitHub: [jsha](#)



**Manish Goregaokar**  
GitHub: [Manishearth](#)



**Wim**  
GitHub: [Nemo157](#)



**Michael Howell**  
GitHub: [notriddle](#)

## Alumni

We also want to thank all past members for their invaluable contributions!



**Jakob Wiesmore**  
GitHub: [CraftSpider](#)



**Jynn Nelson**  
GitHub: [jyn514](#)



**Daniel Silverstone**  
GitHub: [kinnison](#)



**Oliver Middleton**  
GitHub: [ollie27](#)



**Onur Aslan**  
GitHub: [onur](#)



**QuietMisdreavus**  
GitHub: [QuietMisdreavus](#)



**Steve Klabnik**  
GitHub: [steveklabnik](#)

# The Tool

- Build a website with the crate documentation
- List the signatures of all public elements of a crate
- Organized by module
- With search, links

**docs.rs**

# The Website

- You all know it!
- <https://docs.rs>
- In `rust-lang/docs.rs`

# Docs.rs team

Docs.rs, the documentation hosting service for crates

DOCS.RS TEAM REPOSITORY

#T-DOCS-RS ON ZULIP

## Members



**Denis Cornehl**

GitHub: [syphar](#)  
Team leader



**Sebastian Thiel**

GitHub: [Byron](#)



**Guillaume Gomez**

GitHub: [GuillaumeGomez](#)



**Jacob Hoffman-Andrews**

GitHub: [jsa](#)



**Wim**

GitHub: [Nemo157](#)

## Alumni

We also want to thank all past members for their invaluable contributions!



**Jynn Nelson**

GitHub: [jyn514](#)



**Chase Wilson**

GitHub: [Kixiron](#)



**Onur Aslan**

GitHub: [onur](#)



**Pietro Albini**

GitHub: [pietroalbini](#)



**QuietMisdreavus**

GitHub: [QuietMisdreavus](#)

# The Website

- Build the documentation of every crate published on [crates.io](https://crates.io)
- Keep the documentation of every published version
- Use nightly rust
- <https://docs.rs/about>



# Advanced Features

# Let's Start Simple

- `///` for documenting an item

```
/// This is documenting the module  
mod example {}
```

- `//!` for documenting a container

```
mod example {  
    //! This is documenting the module  
}
```

# Code Examples in Documentation

```
/// ```  
/// # fn year() -> u32 {2024}  
/// // Some explanation.  
/// println!("Hello, EuroRust {}!", year());  
/// ```
```

- Hide setup lines with #
- Will be run as doc tests
- Attributes: ignore, should\_panic, no\_run, compile\_fail
- More details in the rustdoc book

# Linking to Items by Name

- Can link to items in scope

```
use std::sync::mpsc::Receiver;

/// This version of [Receiver] supports [future](std::future).
///
/// You can obtain a [Future] by calling [Self::recv()].
/// [Future]: std::future::Future
pub struct AsyncReceiver {
    sender: Receiver
}

impl AsyncReceiver {
    pub async fn recv() -> T { unimplemented!() }
}
```

- More details in the rustdoc book

# Scraping Examples

```
cargo doc -Zunstable-options -Zrustdoc-scrape-examples
```

- Find examples where the item being documented is used

```
[~] pub fn transform_point(&self, point: Vec3) -> Vec3
```

Transforms the given point, applying shear, scale, rotation and translation.

This moves point into the local space of this GlobalTransform.

Examples found in repository ?

```
595         for x in 0..resolution.width {
596             let uvw = (uvec3(x, y, z).as_vec3() + 0.5) * scale - 0.5;
597             let pos = global_transform.transform_point(uvw);
598             let voxel_cube = commands
599                 .spawn(MaterialMeshBundle { ... })
599                 .examples/3d/irradiance_volumes.rs (line 597)
```

- More details in the rustdoc book

# The #Idoc! Attribute

```
/// This is a doc comment.
```

```
#[doc = r"This is a doc comment."]
```

- More details in the rustdoc book

# The #Idoc! Attribute

- Configuration at the crate level

```
#![doc(html_logo_url = "https://example.com/logo.jpg")]
```

- Branding: logo, favicon, ...
- Configuration: doc test options, root html, src

- Configuration on items

```
#[doc(hidden)]  
pub struct InternalDetail;
```

- Hidden, Inline

# Include a File

```
#![doc = include_str!("../README.md")]
```

- Included file will be used as documentation
- Code block will be compiled and tested
- If the markdown file is also rendered outside of rustdoc, pay attention to intra doc links



# Other Macros: document-features

- Automatically document your features: crate **document-features**

```
#![doc = document_features::document_features!()]
```

- Reads the Cargo.toml file, extract the list of features and their comments
- Format it and present it in the documentation

# document-features

```
[features]
default = ["foo"]
#! This comments goes on top

## The foo feature enables the `foo` functions
foo = []

## The bar feature enables the bar module
bar = []

#! ### Experimental features
#! The following features are experimental

## Enable the fusion reactor
##
## ⚠ Can lead to explosions
fusion = []
```

This comments goes on top

- **foo** (*enabled by default*) — The foo feature enables the foo functions
- **bar** — The bar feature enables the bar module

## Experimental features

The following features are experimental

- **fusion** — Enable the fusion reactor
- ⚠ Can lead to explosions

# Aliases

```
#[doc(alias = "haptic", "force", "feedback", "vibration", "vibrate")]  
pub enum Rumble {...}
```

- Helps users find the item

# Conditional Compilation for rustdoc

```
#[cfg(doc)]
```

- Can be used to build part of the code only for documentation
- Can be used to document platforms other than the one doing the build

```
#[cfg(any(windows, doc))]  
pub struct WindowsSpecificThing;
```

# Marking Items as Feature Gated

```
#![feature(doc_cfg)]
```

```
pub struct StructUnderFeature {  
    #[cfg(feature = "enable-this")]  
    #[doc(cfg(feature = "enable-this"))]  
    pub a: i32,  
}
```

## Fields

---

a: i32

Available on **crate feature enable-this** only.

# Marking Items as Feature Gated

- Soon will be automatic: `doc_auto_cfg`
- Forces you to use nightly, unless...

```
#![cfg_attr(docsrs, feature(doc_cfg))]  
  
pub struct StructUnderFeature {  
    #[cfg(feature = "enable-this")]  
    #[cfg_attr(docsrs, doc(cfg(feature = "enable-this")))]  
    pub a: i32,  
}
```

# docs.rs Configuration

```
#[cfg(docsrs)]
```

- In your Cargo.toml:

```
[package.metadata.docs.rs]
```

- Select features, targets
- Pass arguments to cargo, rustc, rustdoc
- More details on <https://docs.rs/about/metadata>

# How about some HTML?

```
/// <div style="background-color:rgb(94.1%, 97.3%, 100.0%); width: 10px; padding  
pub const ALICE_BLUE: Srgba = Srgba::new(0.941, 0.973, 1.0, 1.0);
```

```
pub const ALICE_BLUE: Srgba;
```

[−]





# How about some HTML?

```
/*! documentation  
/*!  
/*! <div class="warning">A big warning!</div>  
/*!  
/*! more documentation
```

[–] documentation



A big warning!

more documentation

# How about some HTML?

- rustdoc options to include additional CSS / HTML
- `--extend-css`
- `--html-in-header`, `--html-before-content`, `--html-after-content`
- Set it up with docsrs config to also be used in the published documentation

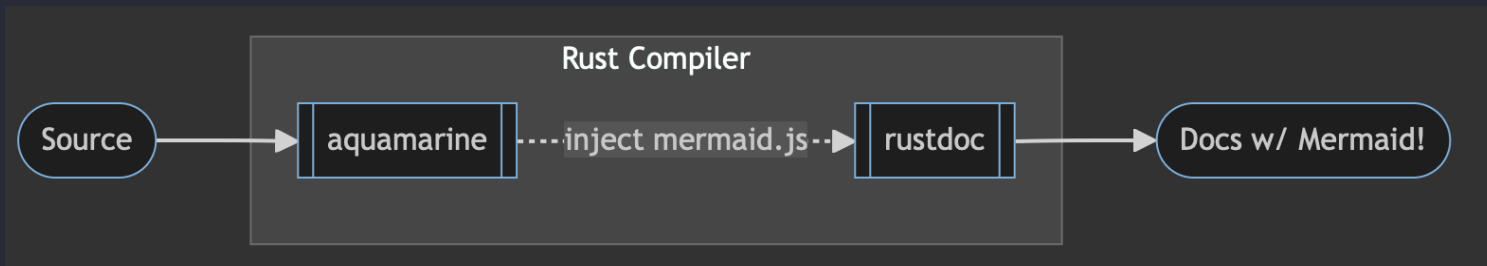
```
[package.metadata.docs.rs]
rustdoc-args = [ "--html-in-header", "path/to/file.html" ]
```

- Be nice!

# Mermaid | Diagramming and Charting Tool

- Crate: **aquamarine**

```
#[cfg_attr(doc, aquamarine)]  
//! ```mermaid  
//! graph LR  
//!     s([Source]) --> a[aquamarine]  
//!     r[rustdoc] --> f([Docs w/ Mermaid!])  
//!     subgraph rustc[Rust Compiler]  
//!         a -. inject mermaid.js .-> r  
//!     end  
//! ```
```



# KaTeX | Math Typesetting Library

- Crate: `katex_doc`

```
/// ```math
/// f(x) = \int_{-\infty}^{\infty} \hat{f}(\xi) e^{2\pi i \xi x} d\xi
/// ```
```

$$f(x) = \int_{-\infty}^{\infty} \hat{f}(\xi) e^{2\pi i \xi x} d\xi$$

**Some things I didn't talk about**

# rustdoc json output interface

- `nightly-rustc/rustdoc_json_types`
- json of everything rustdoc knows about your code
- Can use it for multi steps build
- ... or use your imagination!

# Contributing to rustdoc / docs.rs

- There's always work to do!
- HTML/design? JS/reactivity? Rust introspection? Website management with asynchronous tasks?
- [rustdoc development guide](#)
- [docs.rs git repository](#)

# Some Useful Tips



# Search by Signature

- Searching by type signature for functions

Query	Results
<code>usize -&gt; vec</code>	<code>Vec::with_capacity</code>
<code>vec, vec -&gt; bool</code>	<code>Vec::eq</code>
<code>option&lt;T&gt;, fnonce -&gt; option&lt;U&gt;</code>	<code>Option::map</code>
<code>option&lt;T&gt;, (T -&gt; bool) -&gt; option&lt;T&gt;</code>	<code>Option::filter</code>
<code>iterator&lt;T&gt;, fnmut -&gt; T</code>	<code>Iterator::find</code>

- Use it in links in your documentation

# Implementors Section for Traits

## Implementors

[+] impl SystemParam for &World	source
[+] impl SystemParam for Diagnostics<'_, '_>	source
[+] impl SystemParam for FallbackImageMsaa<'_>	source
[+] impl SystemParam for TransformHelper<'_, '_>	source
[+] impl SystemParam for DefaultUiCamera<'_, '_>	source
[+] impl SystemParam for UiLayoutSystemRemovedComponentParam<'_, '_>	source
[+] impl SystemParam for Commands<'_, '_>	source
[+] impl SystemParam for ParallelCommands<'_, '_>	source

- Use it in links in your documentation

# Documentation Coverage

```
RUSTDOCFLAGS="-Z unstable-options --show-coverage" cargo +nightly doc \
  --workspace --all-features --no-deps
```

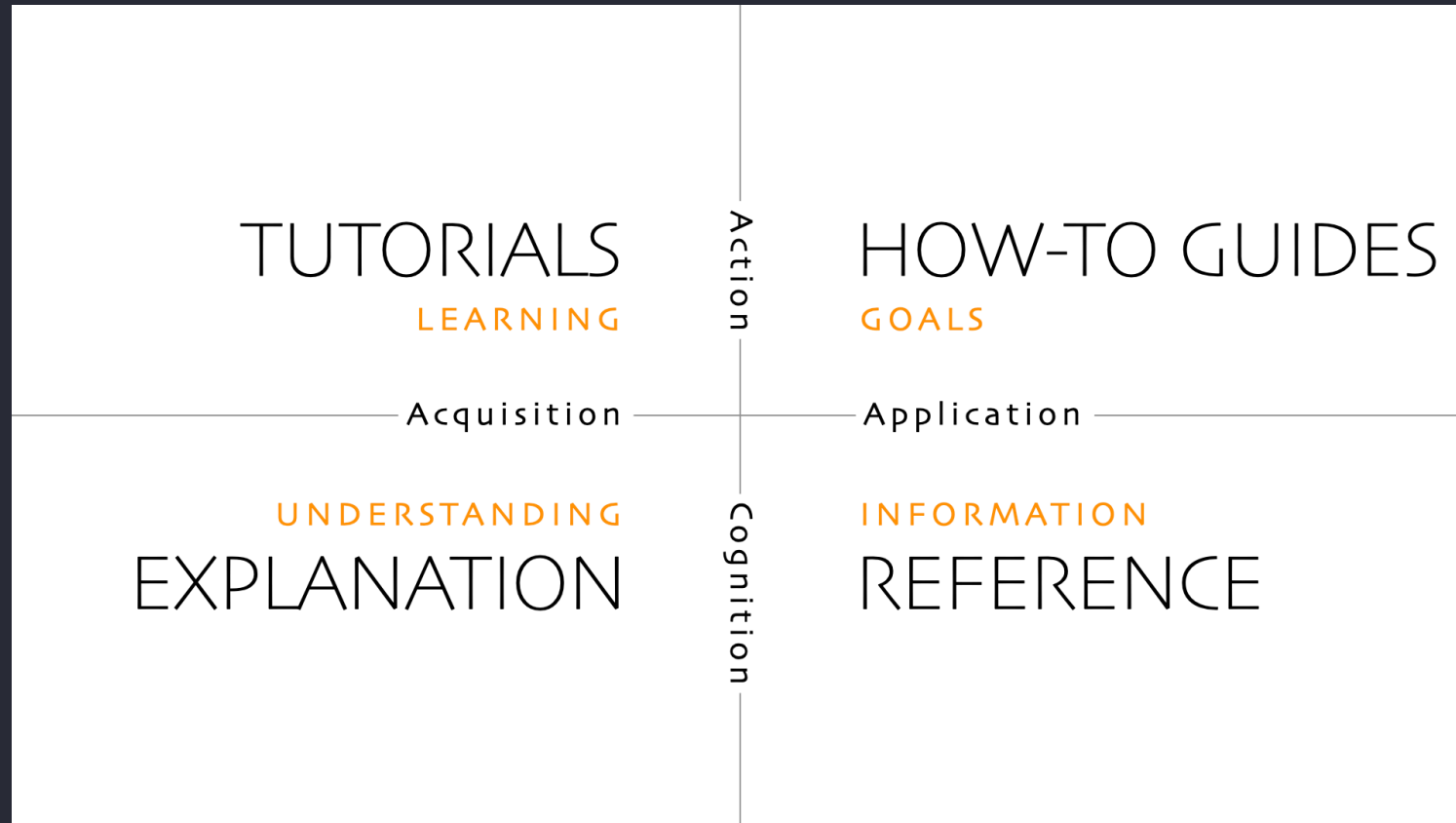
Documenting bevy\_ecs v0.15.0-dev (bevy/crates/bevy\_ecs)

File	Documented	Percentage	Examples	Percentage
crates/bevy_ecs/src/archetype.rs	52	100.0%	0	0.0%
crates/bevy_ecs/src/batching.rs	10	100.0%	0	0.0%
crates/bevy_ecs/src/bundle.rs	18	100.0%	1	5.6%
...				
crates/bevy_ecs/src/world/mod.rs	121	100.0%	30	24.8%
...es/bevy_ecs/src/world/reflect.rs	12	100.0%	1	25.0%
...evy_ecs/src/world/spawn_batch.rs	1	100.0%	0	0.0%
...s/src/world/unsafe_world_cell.rs	41	100.0%	2	4.9%
Total	1001	100.0%	141	24.9%

**So... what is documentation?**

# Rust Guidelines

- [rustdoc book: How to write documentation](#)
- [Rust API Guidelines on Documentation](#)
- [Rust By Example on Documentation](#)



# Diátaxis

- <https://diataxis.fr>
- Content: what to write
- Style: how to write it
- Architecture: how to organize it

# The Good Docs Project

- <https://www.thegooddocsproject.dev>
- Free templates for many kind of documentation documents
- Avoid the blank page syndrome



**Where does it fit?**

# docs

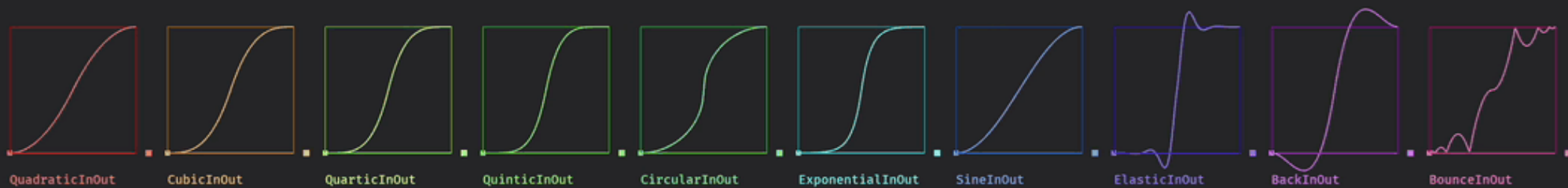
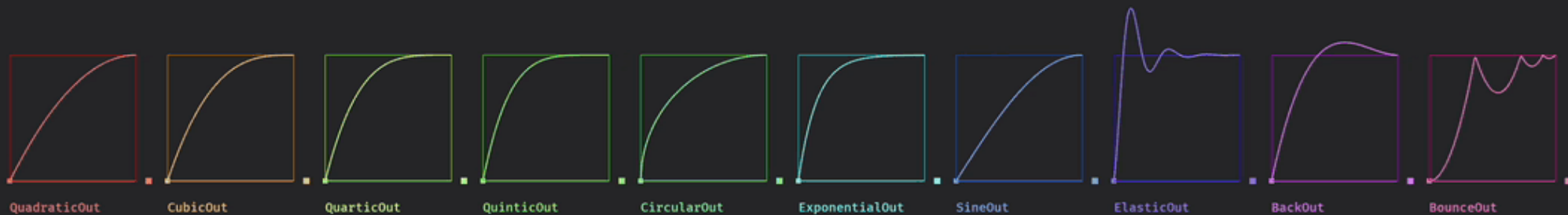
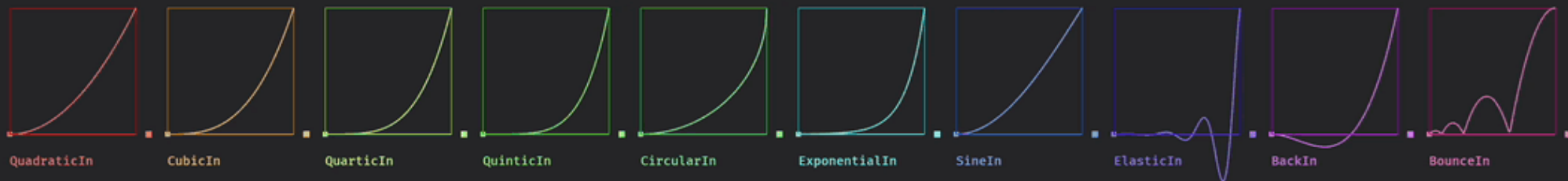
- In the reference corner
- Best when a user is already using your crate
- Need to quickly find a specific information
- Focus on precision, linking related items, easy search

# Examples

- Examples are in the How-To Guides corner
- Focus on how to use the crate to solve a use case
- Bridge from docs by scraped examples

Variants
<div>QuadraticIn</div> <div>f(t) = t²</div>
<div>QuadraticOut</div> <div>f(t) = -(t * (t - 2.0))</div>
<div>QuadraticInOut</div> <div>Behaves as EaseFunction::QuadraticIn for t&lt;0.5 and as EaseFunction::QuadraticOut for t&gt;=0.5</div>
<div>CubicIn</div> <div>f(t) = t³</div>
<div>CubicOut</div> <div>f(t) = (t - 1.0)³ + 1.0</div>
<div>CubicInOut</div> <div>Behaves as EaseFunction::CubicIn for t&lt;0.5 and as EaseFunction::CubicOut for t&gt;=0.5</div>
<div>QuarticIn</div> <div>f(t) = t⁴</div>
<div>QuarticOut</div> <div>f(t) = (t - 1.0)³ * (1.0 - t) + 1.0</div>
<div>QuarticInOut</div> <div>Behaves as EaseFunction::QuarticIn for t&lt;0.5 and as EaseFunction::QuarticOut for t&gt;=0.5</div>
<div>QuinticIn</div> <div>f(t) = t⁵</div>
<div>QuinticOut</div> <div>f(t) = (t - 1.0)⁵ + 1.0</div>
<div>QuinticInOut</div> <div>Behaves as EaseFunction::QuinticIn for t&lt;0.5 and as EaseFunction::QuinticOut for t&gt;=0.5</div>
<div>SineIn</div> <div>f(t) = 1.0 - cos(t * π / 2.0)</div>
<div>SineOut</div> <div>f(t) = sin(t * π / 2.0)</div>
<div>SineInOut</div> <div>Behaves as EaseFunction::SineIn for t&lt;0.5 and as EaseFunction::SineOut for t&gt;=0.5</div>

<div>CircularIn</div> <div>f(t) = 1.0 - sqrt(1.0 - t²)</div>
<div>CircularOut</div> <div>f(t) = sqrt((2.0 - t) * t)</div>
<div>CircularInOut</div> <div>Behaves as EaseFunction::CircularIn for t&lt;0.5 and as EaseFunction::CircularOut for t&gt;=0.5</div>
<div>ExponentialIn</div> <div>f(t) = 2.0^(10.0 * (t - 1.0))</div>
<div>ExponentialOut</div> <div>f(t) = 1.0 - 2.0^(-10.0 * t)</div>
<div>ExponentialInOut</div> <div>Behaves as EaseFunction::ExponentialIn for t&lt;0.5 and as EaseFunction::ExponentialOut for t&gt;=0.5</div>
<div>ElasticIn</div> <div>f(t) = -2.0^(10.0 * t - 10.0) * sin((t * 10.0 - 10.75) * 2.0 * π / 3.0)</div>
<div>ElasticOut</div> <div>f(t) = 2.0^(-10.0 * t) * sin((t * 10.0 - 0.75) * 2.0 * π / 3.0) + 1.0</div>
<div>ElasticInOut</div> <div>Behaves as EaseFunction::ElasticIn for t&lt;0.5 and as EaseFunction::ElasticOut for t&gt;=0.5</div>
<div>BackIn</div> <div>f(t) = 2.70158 * t³ - 1.70158 * t²</div>
<div>BackOut</div> <div>f(t) = 1.0 + 2.70158 * (t - 1.0)³ - 1.70158 * (t - 1.0)²</div>
<div>BackInOut</div> <div>Behaves as EaseFunction::BackIn for t&lt;0.5 and as EaseFunction::BackOut for t&gt;=0.5</div>
<div>BounceIn</div> <div>bouncy at the start!</div>
<div>BounceOut</div> <div>bouncy at the end!</div>
<div>BounceInOut</div> <div>Behaves as EaseFunction::BounceIn for t&lt;0.5 and as EaseFunction::BounceOut for t&gt;=0.5</div>



Progress: 0.00

# Explanation and Tutorials

## Included markdown files

- Can be referenced without building docs
- Easier for long form text

## Documentation only modules

- Modules without code, only documentation
- See [clap documentation](#) for an example (`_tutorial`, `_faq`, ...)

# mdBook

- [mdBook documentation](#)
- Easy to build books from markdown
- Rust code examples are tested
- Natural progression from rustdoc
- Widely used in the Rust ecosystem

# Static Site Generators

- Cobalt, Zola
- Freeform website generators from markdown files
- No code validation\*



**Write good docs!**