README 的道术

以 TM 为例

1 项目名称

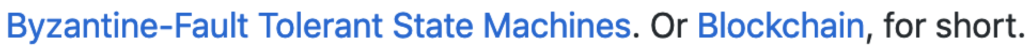
图 1-1

Tendermint

2 相关重要概念的解释

2.1 道

图 1-2



Byzantine-Fault Tolerant State Machines. Or Blockchain, for short.

图 1-2 中 `Byzantine-Fault Tolerant` `State Machines` `Blockchain` 这几个 TM 中涉及到的重要概念都直接链接到了 wiki

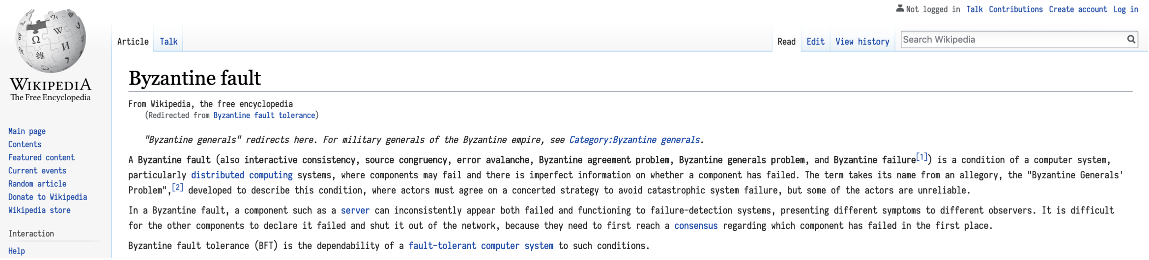
2.2 术

markdown 链接

代码 1-1

```
[Byzantine-Fault Tolerant](https://en.wikipedia.org/wiki/Byzantine_fault_tolerance)
```

图 1-3



Byzantine fault

From Wikipedia, the free encyclopedia
(Redirected from Byzantine fault tolerance)

"Byzantine generals" redirects here. For military generals of the Byzantine empire, see Category:Byzantine generals.

A **Byzantine fault** (also **interactive consistency**, **source congruency**, **error avalanche**, **Byzantine agreement problem**, **Byzantine generals problem**, and **Byzantine failure**[1]) is a condition of a computer system, particularly distributed computing systems, where components may fail and there is imperfect information on whether a component has failed. The term takes its name from an allegory, the "Byzantine Generals' Problem",[2] developed to describe this condition, where actors must agree on a concerted strategy to avoid catastrophic system failure, but some of the actors are unreliable.

In a Byzantine fault, a component such as a server can inconsistently appear both failed and functioning to failure-detection systems, presenting different symptoms to different observers. It is difficult for the other components to declare it failed and shut it out of the network, because they need to first reach a consensus regarding which component has failed in the first place.

Byzantine fault tolerance (BFT) is the dependability of a fault-tolerant computer system to such conditions.

图 1-3 显示了 **拜占庭容错** 的 **wiki**

3 Badge

3.1 道

图 1-4

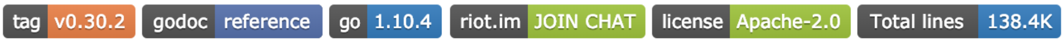


图 1-4 中有若干标签，这些标签简明扼要的给出了一些信息。版本，文档，语言，论坛，许可证，代码。其他的项目也可能别的标签，比如编译。

3.2 术

代码 1-2

```
[![Go version](https://img.shields.io/badge/go-1.10.4-blue.svg)](https://github.com/moovweb/gvm)
```

代码 1-2 给出了 go 1.10.4 的实现方式，是 **链接嵌套图片**。

1. 链接的地址是 **gvm** 项目 (一个允许多个不同版本的 golang 同时存在，能自由切换的工具)，显示的是一个 **svg** 图片
2. **svg** 图片的生成多使用 shields.io [https://shields.io/]

3.2.1 shields.io 的简单使用

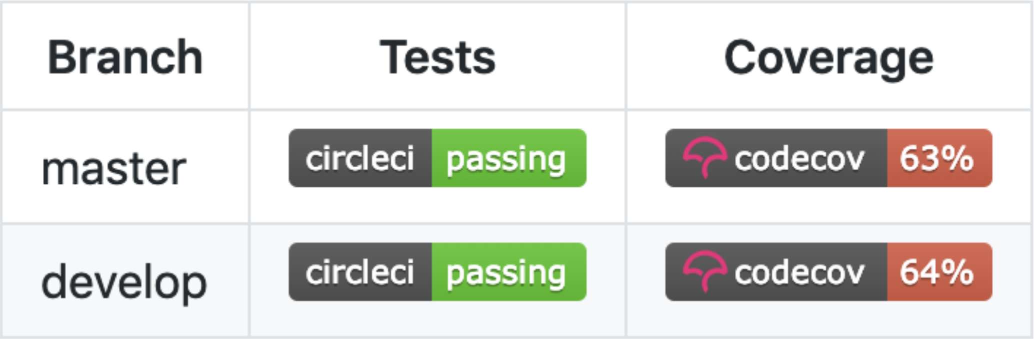
`https://img.shields.io/badge/` 后面跟 **name-value-color**.svg

不用事先生成，可直接使用

4 测试情况

4.1 道

图 1-5



Branch	Tests	Coverage
master	circleci passing	codecov 63%
develop	circleci passing	codecov 64%

图 1-5 给出了不同分支的测试及覆盖率情况

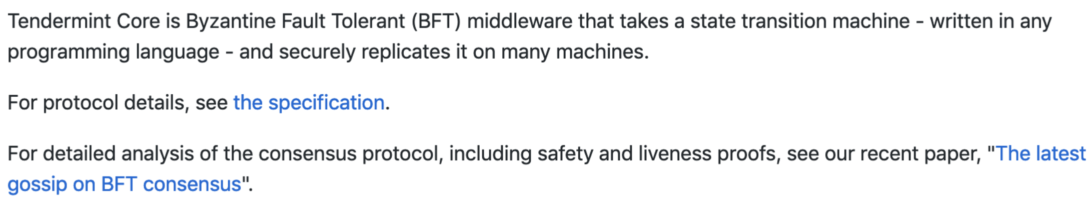
4.2 术

1. 表格嵌套 Badge
2. 使用了 circleci [https://circleci.com], codecov [https://codecov.io/] 两个工具，**具体使用方式现在不明，有待研究**

5 概要描述

5.1 道

图 1-6



Tendermint Core is Byzantine Fault Tolerant (BFT) middleware that takes a state transition machine - written in any programming language - and securely replicates it on many machines.

For protocol details, see the specification.

For detailed analysis of the consensus protocol, including safety and liveness proofs, see our recent paper, "The latest gossip on BFT consensus".

图 1-6 对项目进行简短的描述

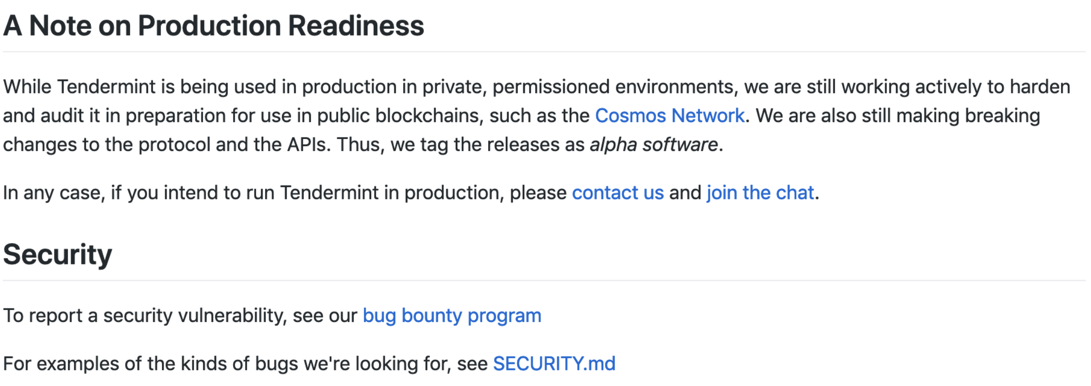
5.2 术

1. 几句话说出 **这个项目是什么**
2. 相关的重要概念或内容可以给出链接

6 插入的重要说明

6.1 道

图 1-7



A Note on Production Readiness

While Tendermint is being used in production in private, permissioned environments, we are still working actively to harden and audit it in preparation for use in public blockchains, such as the Cosmos Network. We are also still making breaking changes to the protocol and the APIs. Thus, we tag the releases as *alpha software*.

In any case, if you intend to run Tendermint in production, please contact us and join the chat.

Security

To report a security vulnerability, see our bug bounty program

For examples of the kinds of bugs we're looking for, see SECURITY.md

图 1-7 中的两个说明应该是 TM 项目比较重视的两点，不具有共性。

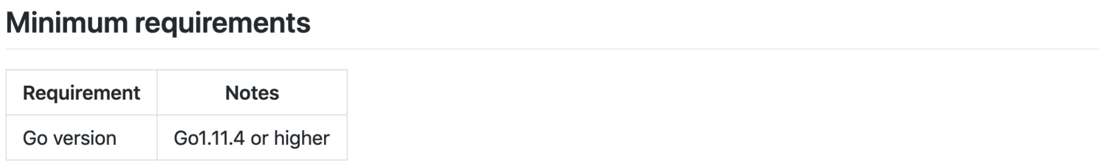
6.2 术

根据项目特点，把比较重要的，想要强调的部分放在靠前的位置。

7 项目依赖

7.1 道

图 1-8



Minimum requirements

Requirement	Notes
Go version	Go1.11.4 or higher

图 1-8 给出了项目的依赖，让读者可以明确项目使用的 环境，语言，库 及其版本的信息。

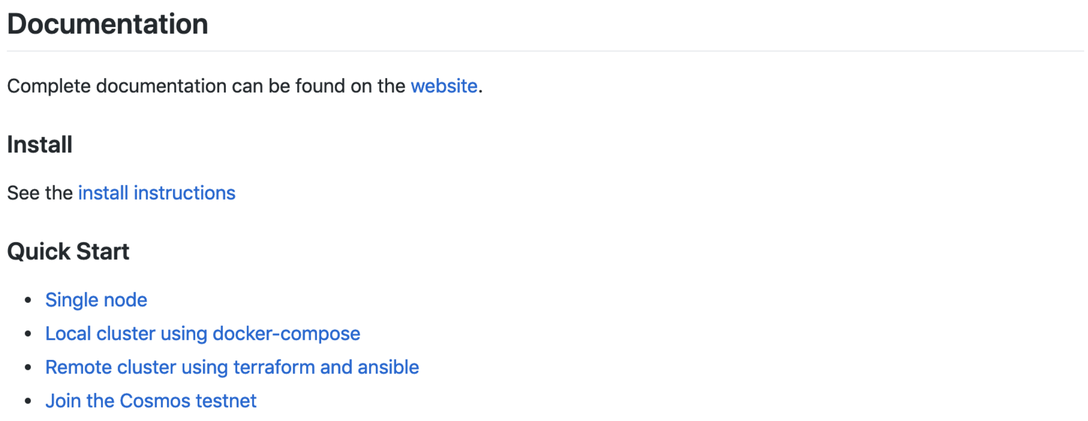
7.2 术

表格的方式给出

8 文档

8.1 道

图 1-9



Documentation

Complete documentation can be found on the website.

Install

See the install instructions

Quick Start

- Single node
- Local cluster using docker-compose
- Remote cluster using terraform and ansible
- Join the Cosmos testnet

图 1-9 给出了文档的相关信息。这里是读者了解项目最重要的地方。它分成了三个部分:

1. website 是完整文档
2. Install 给出安装说明
3. Quick Start 给出了使用示例

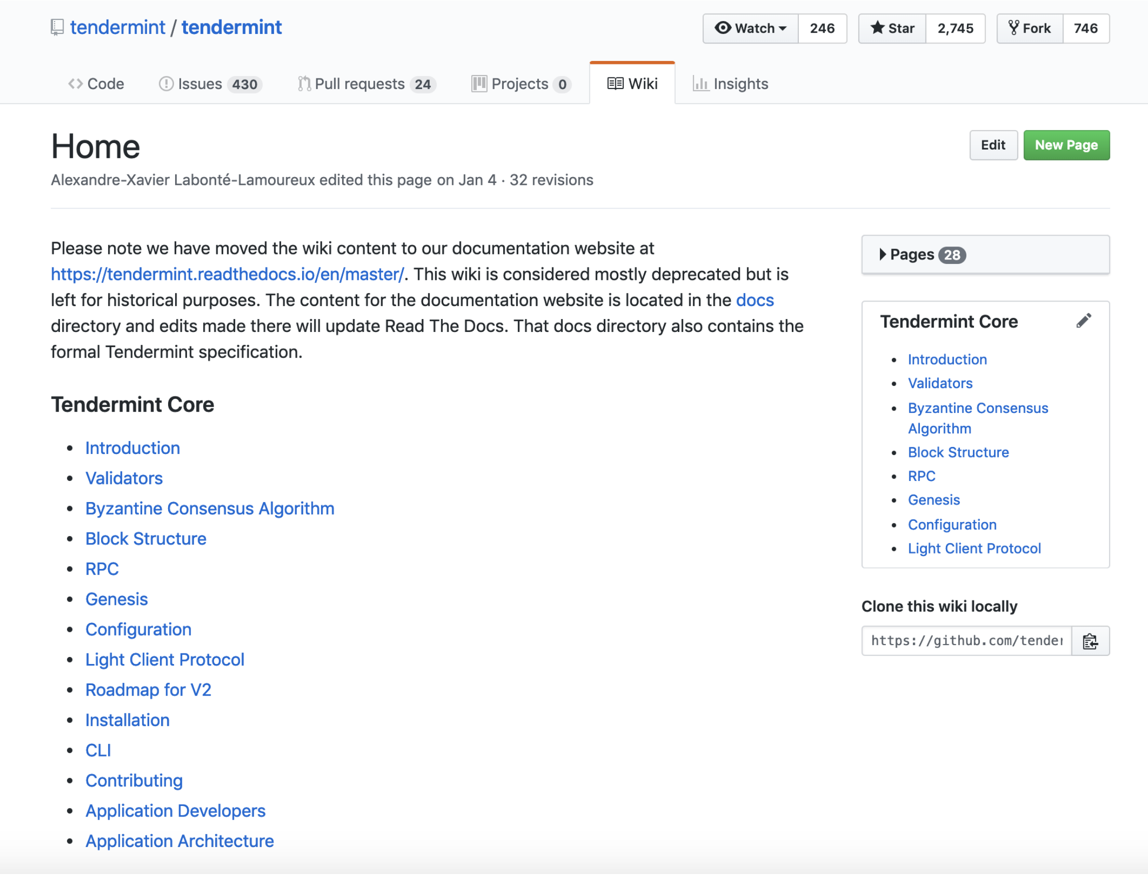
8.1 术

8.1.1 website

TM 的这个 website 是用 VuePress [https://vuepress.vuejs.org/] 做的，把 markdown 文档编译成了静态网站，这一点跟 MkDocs [https://www.mkdocs.org/] 很像。Github Pages [https://pages.github.com/] 是 github 提供的一种展示文档的方式，可以把编译好的静态网站宿主在上面，但是要求 **项目必须是 public 的**

展示大量信息的另一种方式是 github 的 **wiki**，wiki 跟项目不在同一个 repository，需要单独维护。跟 website 相比，readme, wiki 的缺点是 **只支持最基本的 markdown 语法**，而 website，比如 MkDocs 就支持插件，如 mermaid，plantuml, admonition 等一系列插件，更方便文档的维护，也使得文档的表现力更强。

图 1-10



Home

Alexandre-Xavier Labonté-Lamoureux edited this page on Jan 4 · 32 revisions

Please note we have moved the wiki content to our documentation website at https://tendermint.readthedocs.io/en/master/. This wiki is considered mostly deprecated but is left for historical purposes. The content for the documentation website is located in the docs directory and edits made there will update Read The Docs. That docs directory also contains the formal Tendermint specification.

Tendermint Core

- Introduction
- Validators
- Byzantine Consensus Algorithm
- Block Structure
- RPC
- Genesis
- Configuration
- Light Client Protocol
- Roadmap for V2
- Installation
- CLI
- Contributing
- Application Developers
- Application Architecture

Tendermint Core

- Introduction
- Validators
- Byzantine Consensus Algorithm
- Block Structure
- RPC
- Genesis
- Configuration
- Light Client Protocol

图 1-10 是 TM 的 wiki

图 1-11

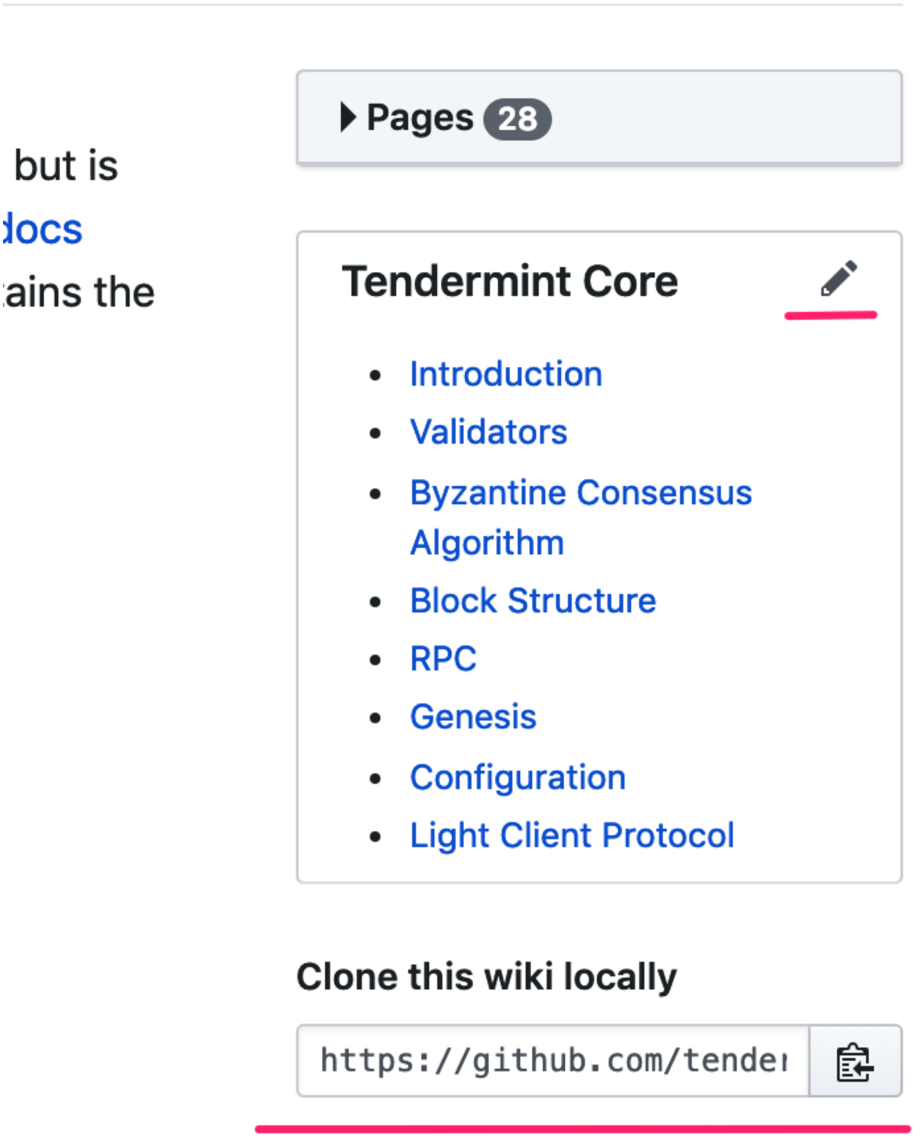


图 1-11 显示了 wiki 可以 **在线编辑**，也可以 **本地编辑**

从地址 `https://github.com/tendermint/tendermint.wiki.git` 可看出 wiki 与 TM 项目分属于不同 repository

> **wiki 与 website 的区别**
>
> 1. wiki 中的流程图，时序图，干特图 只能放图片，如果有改动，只能重新结截图; website 的各种图由插件生成，有专门的 markdown 语法，修改起来方便
> 2. wiki 样式相对固定; website 的样式相对灵活
> 3. wiki 简单易用; website 门槛略高，需要学习工具的使用。

8.1.2 Install

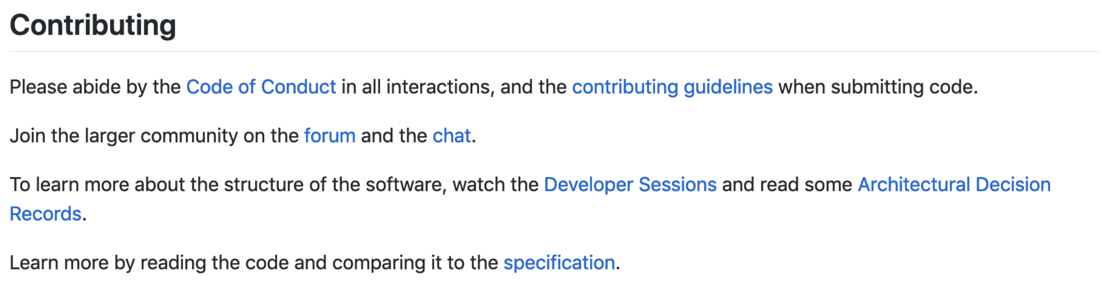
Install instructions 链接到了另一个文件， 里面介绍了: **如何安装，如何运行，如何重装** 等信息。文内有相关说明及示例代码。

8.1.3 Quick Start

Quick Start 给出了 4 个链接，指向了 4 个不同位置。前 3 个指向了文件，最后一个指向了 Cosmos 的文档。

9 贡献

图 1-12



Contributing

Please abide by the Code of Conduct in all interactions, and the contributing guidelines when submitting code.

Join the larger community on the forum and the chat.

To learn more about the structure of the software, watch the Developer Sessions and read some Architectural Decision Records.

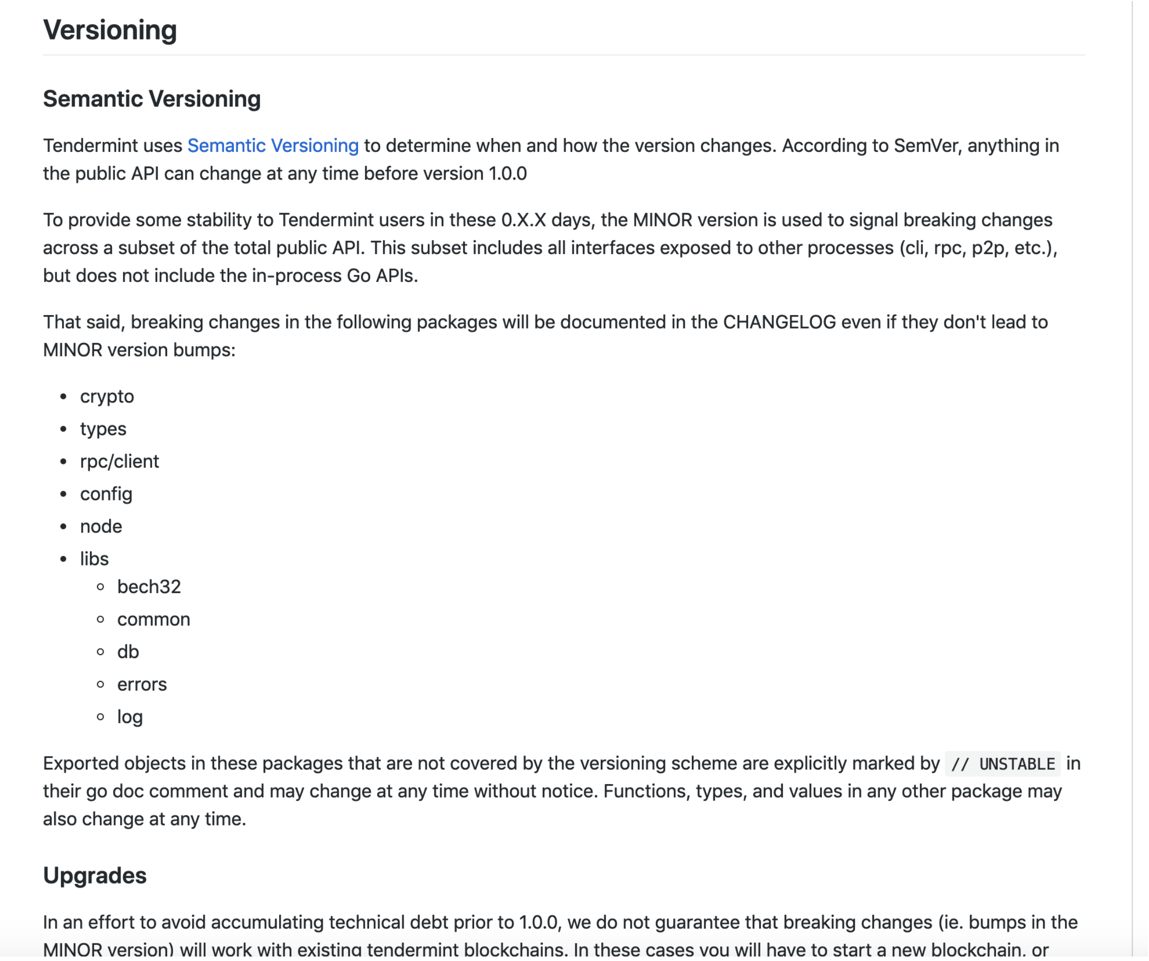
Learn more by reading the code and comparing it to the specification.

图 1-12 主要是介绍如何参与项目。这一段虽短，但给出了多个链接:

1. Code of Conduct 是行为规范
2. contributing guidelines 是参与流程
3. forum 和 chat 是贡献者之前的沟通方式
4. Develper Sessions 是放到 youtube 上的视频 (没看内容)
5. Architetural Decision Records 是架构设计
6. specification 是整个代码结构说明

10 版本

图 1-13



Versioning

Semantic Versioning

Tendermint uses Semantic Versioning to determine when and how the version changes. According to SemVer, anything in the public API can change at any time before version 1.0.0

To provide some stability to Tendermint users in these 0.X.X days, the MINOR version is used to signal breaking changes across a subset of the total public API. This subset includes all interfaces exposed to other processes (cli, rpc, p2p, etc.), but does not include the in-process Go APIs.

That said, breaking changes in the following packages will be documented in the CHANGELOG even if they don't lead to MINOR version bumps:

- crypto
- types
- rpc/client
- config
- node
- libs
  - bech32
  - common
  - db
  - errors
  - log

Exported objects in these packages that are not covered by the versioning scheme are explicitly marked by `// UNSTABLE` in their go doc comment and may change at any time without notice. Functions, types, and values in any other package may also change at any time.

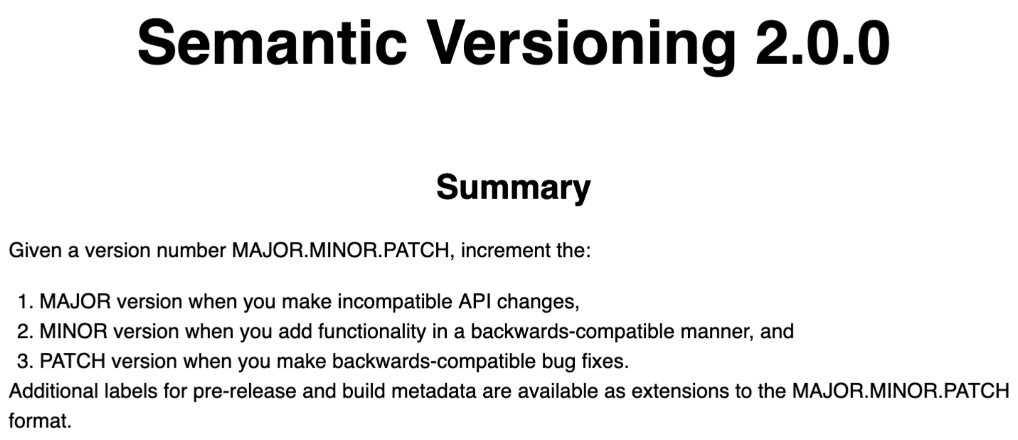
Upgrades

In an effort to avoid accumulating technical debt prior to 1.0.0, we do not guarantee that breaking changes (ie. bumps in the MINOR version) will work with existing tendermint blockchains. In these cases you will have to start a new blockchain, or

图 1-13 给出了项目更新与版本号的关系。TM 像很多项目一样，也使用了 Semantic Versioning [https://semver.org/]

打开链接，可以看到

图 1-14



Semantic Versioning 2.0.0

Summary

Given a version number MAJOR.MINOR.PATCH, increment the:

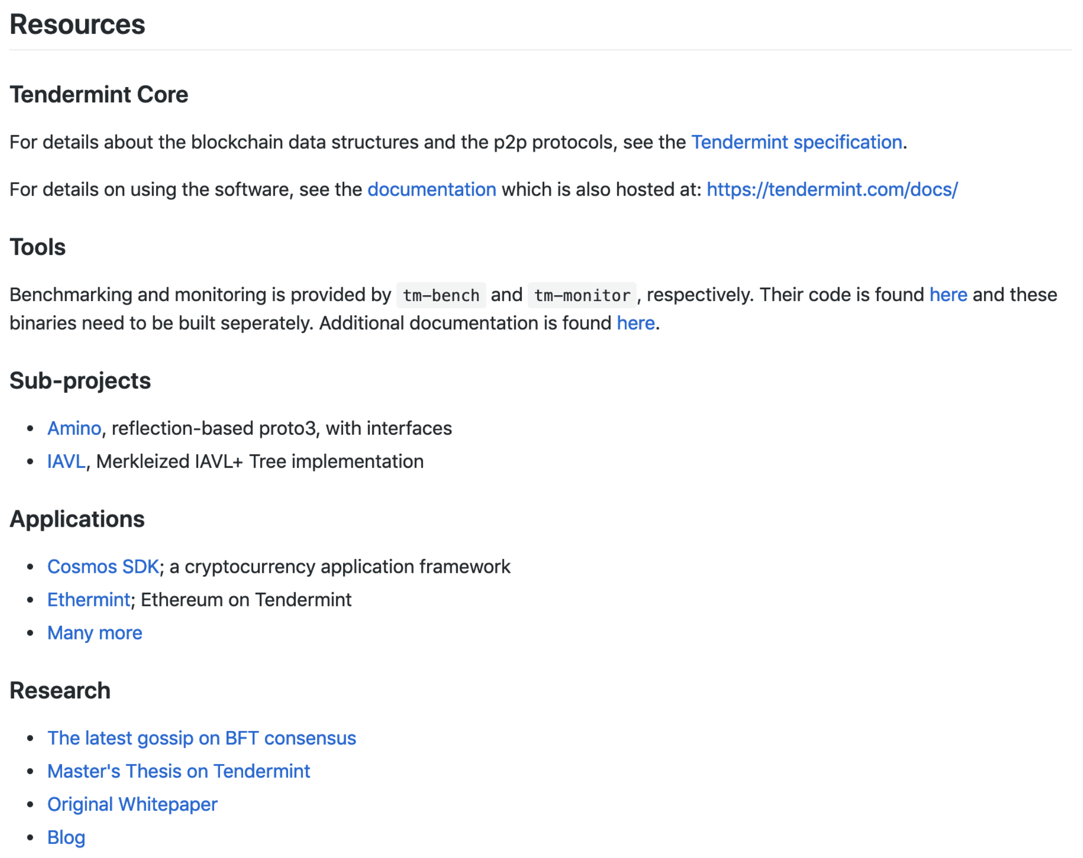
1. MAJOR version when you make incompatible API changes,
2. MINOR version when you add functionality in a backwards-compatible manner, and
3. PATCH version when you make backwards-compatible bug fixes.

Additional labels for pre-release and build metadata are available as extensions to the MAJOR.MINOR.PATCH format.

图 1-14 给出了版本号定义的基本规则

11 资源

图 1-15



Resources

Tendermint Core

For details about the blockchain data structures and the p2p protocols, see the Tendermint specification.

For details on using the software, see the documentation which is also hosted at: https://tendermint.com/docs/

Tools

Benchmarking and monitoring is provided by `tm-bench` and `tm-monitor`, respectively. Their code is found here and these binaries need to be built seperately. Additional documentation is found here.

Sub-projects

- Amino, reflection-based proto3, with interfaces
- IAVL, Merkleized IAVL+ Tree implementation

Applications

- Cosmos SDK; a cryptocurrency application framework
- Ethermint; Ethereum on Tendermint
- Many more

Research

- The latest gossip on BFT consensus
- Master's Thesis on Tendermint
- Original Whitepaper
- Blog

图 1-15 给出了其他相关的一些信息。算是一些补充。

会议记要

> ⚠ **3 Badge**
>
> tag, go, python, license 必须有，godoc 争取有; badge 可以没有链接

> ⚠ **4 测试**
>
> 至少两个分支，可以没有链接，数据由 台帅 给出

5 概要描述

可以链接到 白皮书

6 插入的重要说明

1. 统一联系方式
2. bug 提交方式，邮件
3. 简要说明，用户有哪些限制

7 项目依赖

操作系统，语言，库 都要写

8 文档

1. website 可以没有，或链接到 documentation 项目 (飞哥提出)
2. Install 必须要有
3. Quick Start 必须要有，不必复杂
4. 操作系统可以先只写一种，比如 mac

9 贡献

1. **开发者指南** 有统一模板
2. **架构设计** 要说明原理

11 资源

统一链接到 **白皮书** 等资源，资源由 明哥 给出