My Docs 2

None

None

None

Table of contents

1. Intro

1.0.1 Постановка задачи и концепция решения

Практика docs-as-code выглядит потенциально полезной для ведения проектной и продуктовой документации.

Цель проекта попробовать подход docs-as-code перед тем как предлагаеть его использование в команде. Опробирование включает в себя запуск MVP небходимой инфрастукруты, описание процессов работы, публикацию примера документации.

В качестве языка разметки был выбран markdown ввиду простоты его освоения и наличия WYSIWIG редакторов. В роли WYSIWIG редактора был выбран Typora, одним из премуществ которого является удобный редактор markdown таблиц и нативная поддержка диаграм mermaid.

Mkdocs-material выбран для сборки и публикации документации т.к. он обладает минимально достаточным функционалом и его просто развернуть на GitHub.

2. Test document in markdown

2.0.1 Примеры медиа вставок

Таблица

Чтобы комфортно работать с таблицами в markdown, нужен редактор который поддерживает WYSIWYG редактирование таблиц. Например, Typora или Obsidian. Тем не менее функционал работы с таблицами беден в сравнении с notion или coda. Таблица в markdown это как простая таблица в notion в которой нельзя делать ни сортировку, ни ссылки на другие таблицы.

Колонка 1	Колонка 2	Колонка 3
Ряд 1		
Ряд 2		
Ряд 3		

Картинка

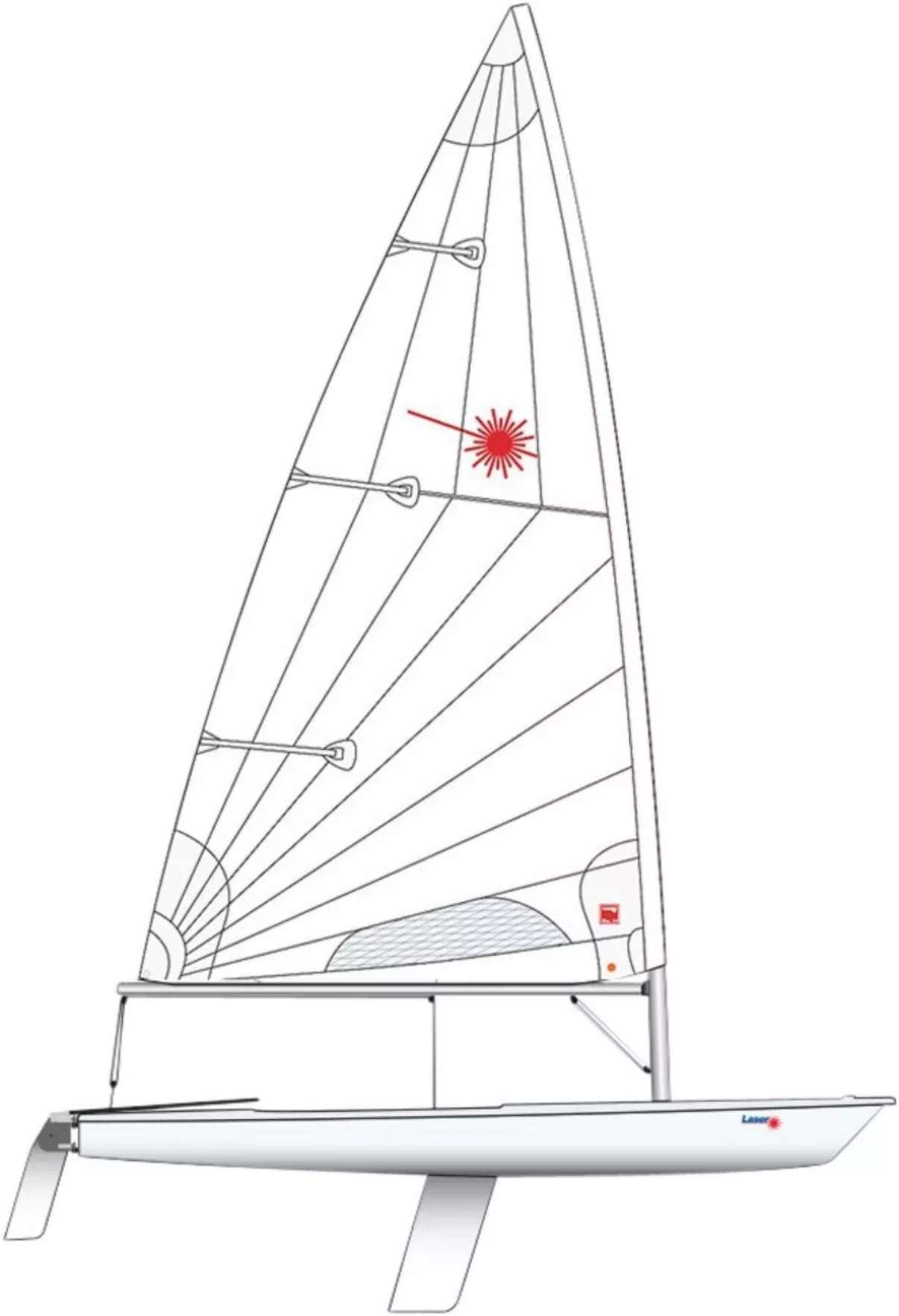
Laser

Диаграмма mermaid

```
sequenceDiagram
  autonumber
  Alice->>John: Hello John, how are you?
  loop Healthcheck
      John->>John: Fight against hypochondria
  end
  Note right of John: Rational thoughts!
  John-->>Alice: Great!
  John->>Bob: How about you?
  Bob-->>John: Jolly good!
```

3. Как вести разработку

3.0.1 Как меняется процесс разработки документации, например, технического задания.

Как было, если использовать google docs или notion

1. Завел задачу на доработку документации
2. Открыл документ
3. Написал текст
4. Закрыл задачу на доработку документации

Как будет с docs-as-code с использованием механизма pull-request

1. Завел задачу на доработку документации в github
2. Завел отдельную ветку
3. Сделал pull этой ветки на локальный компьютер
4. Открыл документ в typora
5. Написал текст
6. Сделал коммит
7. Сделал pull реквест
8. Ревьюверы посмотерли пулл реквест
9. Если одобрили, то пулл реквест был принят. Если не одобрили, то текст ушел на доработку.
10. Закрыл задачу
11. Доработанный текст добавили в релиз

Как будет с docs-as-code без использования механизма pull-request

1. Завел задачу на доработку документации в github
2. Открыл документ в typora
3. Написал текст
4. Сделал коммит
5. Получить комментарии от ревьювера
6. Закрыл задачу
7. Доработанный текст добавили в релиз

Список инструкций, которые нужно написать

- Как писать тексты
- Как добавлять таблицы
- Как добавлять картинки
- Как добавлять диаграммы
- Как развернуть инструментарий на github с помощью GitHub pages

4. Цель этого документа

Гипотеза: подход documentation as code к разработке документации (о какой документации идет речь) при разработке софта может оказаться более удобным, чем использование confluence или google doces. Вариант решения - Typora + Mkdocs. Typora - WYSIWYG редактор. Mkdocs - инструментарий для публикации.

Одним из основных плюсов может быть возможность организации процесса ревью с помощью с помощью pull реквестов, что может позволить поставить тиражируемую и контролируемую практику разработки документации - концепции использования, концепции системы, архитектуры,

Если все наработки добавлять в отдельную ветку каждый день и раз в пару дней делать pull реквест в мастер.

Как вести разработку