



OSS Gate ワークショップ

アイスブレイク



✓ 目的

- ✓ 周りの人と話しやすくなること
せっかくだから相談しよう！

✓ やること

- ✓ 参加目的をみんなに説明
※内容より**声を出すことが大事！**
- ✓ まずはサポーターから発声
順番はスムーズな進行のために指定しているだけ。

チャット：Element



- ✓ <https://app.element.io/#/room/#oss-gate:matrix.org>
- ✓ アナウンス：oss-gate/announce
- ✓ 今後もみんなとつながってほしい！
- ✓ まず登録して挨拶しよう！
- ✓ 終わったら今日の成果を自慢しよう！

OSS Gate ?



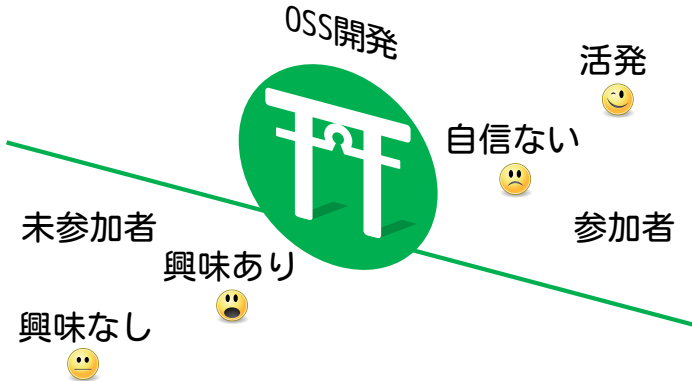
OSSの門？

門

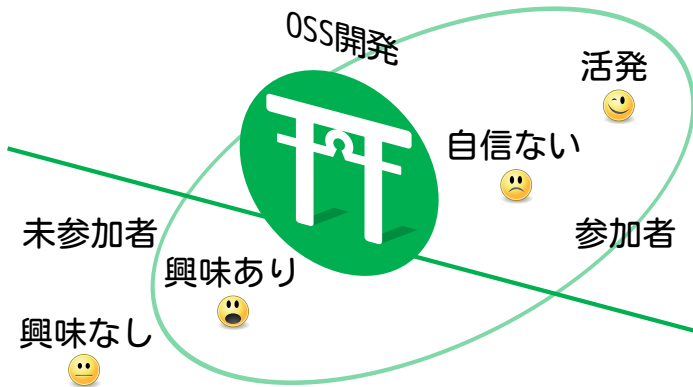


境界にあるもの

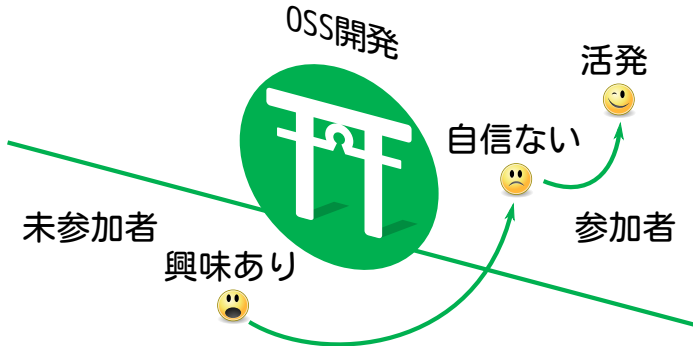
扱う境界



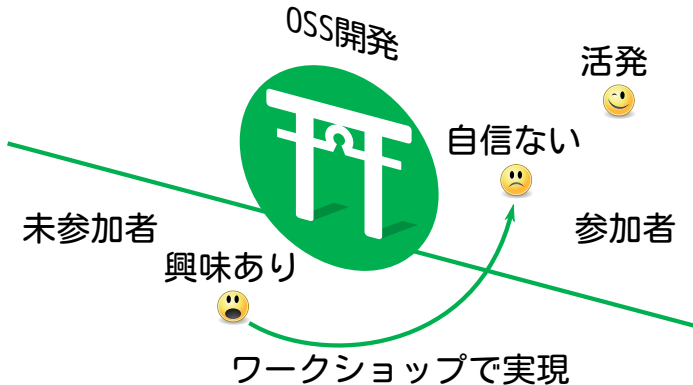
ターゲット



やりたいこと



未参加者→参加者



OSS Gateとワークショップ



- ✓ OSS Gate
 - ✓ [OSS開発参加者を継続的に増やす]
取り組み
- ✓ OSS Gateワークショップ
 - ✓ [...増やす]を実現するための1手段
 - ✓ **未**経験者が経験者になると増える

どうして**未**経験？

- ✓ (数人の参加者に聞く)
- ✓ 予想：
 - ✓ やったことがないから
なんとなく敷居が高いと感じる

ワークショップの重要事項



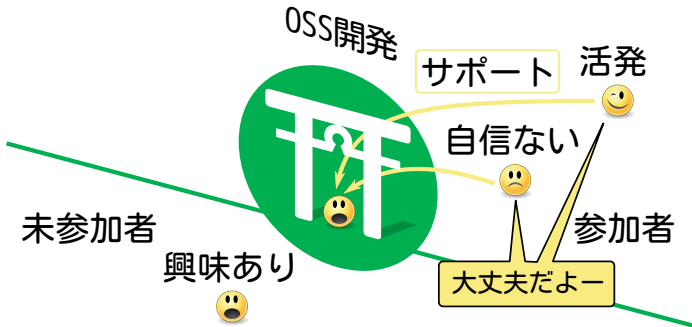
体験する

体験して大したことはないとわかる→敷居が下がる

体験して門をくぐる



よりくぐりやすく



ワークショップの内容1



参加者のこと

立場一覧

- ✓ ビギナー
- ✓ サポーター
- ✓ サポートメンター
- ✓ 進行役

ビギナー

- ✓ OSSの開発に参加したい
 - ✓ でも参加したことはない
- ✓ OSSの開発に参加した事はある
 - ✓ でもまだ自信がない

サポーター



- ✓ ビギナーのサポート係
- ✓ OSS開発経験者
- ✓ 初参加でも大丈夫！
 - ✓ 例：進行役がやることを随時説明
 - ✓ 例：サポートメンターがサポート

サポートメンター



- ✓ サポーターのサポート係
- ✓ サポーター経験者
- ✓ 会場各地でスポットサポート
- ✓ サポート例：
 - ✓ うまくサポートできていない感…
→相談しよう！
サポーター1人で完璧にサポートしなくてよい！

進行役



- ✓ 進行と全体を気にかける係

ワークショップの内容2



流れ

今日の流れのポイント



- ✓ 未経験者の最初の1歩に最適化
 - ✓ ※OSSの開発方法はいろいろある
 - ✓ ※やりたい事がある人は応相談
 - ✓ ※基本的にこのやり方でやろう！

流れ

1. ユーザーとしてOSSを動かす
2. ↑で気づいた事を開発元に
フィードバック

期待

- ✓ 普段は気づいていないだけで
実はフィードバックポイントが
あったことを**体験**して！
 - ✓ ※ググったり生成AIに聞いて回避していない？
そんなときどうしたらよいかはワークショップ内で！
- ✓ フィードバックを**体験**して！

ワークショップの内容3



ユーザーとして
OSSを動かす

動かす流れ

(詳細は後述)

1. 対象OSSを決める
2. 作業メモを書く場所を用意
3. 作業メモを書きながら
公式サイト・README通り
動かす

OSSとは

- ✓ オープンソースライセンスを設定したソフトウェア

- ✓ <https://opensource.org/licenses/>

- ✓ ライセンスを確認すればOSSかどうかわかる

- ✓ OSS「っぽい」は存在しない

対象OSS決め

- ✓ ビギナーが決める
 - ✓ 使っているOSSから選ぶ
ブラウザの拡張機能・便利なコマンドラインツール等
無意識で使っている物の中にもOSSはある
 - ✓ 難易度は気にしなくてよい！
サポーターがサポートするから！
- ✓ サポーターは↑をサポート
 - ✓ 自分の知らないOSSや言語でもよい
ビギナーと一緒に悩んであげよう！

対象OSS決めデモ




デモ

- ✓ 最近使っているOSSは？
 - ✓ ライセンス確認→OK！
- ✓ その中で一番ときめくのは？
- ✓ ではそれにしましょう！

動かすときのポイント



- ✓ 作業メモを書く
 - ✓  メモを書く場所はこのあと作る
- ✓ なにかする毎に書く
 - ✓ 例：ドキュメントを読み始めた
 - ✓ 例：次のドキュメントを読み始めた

作業メモを書く場所を作る



デモ

1. GitHub: `oss-gate/workshop`
2. ↑にissueを作る
3. 周囲のビギナーの人たちが作ったissueにコメント

作業メモの例



ドキュメント通りインストールしたけど
失敗した。

よりよい作業メモの例




https://... のインストール手順をなぞろう！
(↑後から再度参照できるようにURLも書く)
brewでインストールできるはずなのに失敗した
(↑期待する結果)

```
$ brew install XXX (←なにをしたか)  
(...コマンドの実行結果...)  
(↑実際の結果)
```

```
XXX is not found
```

↑というようにパッケージがないと言われる



ユーザーとして動かす デモ

1. 公式サイトを開く
2. 作業メモを書く
3. 概要を読む
4. 作業メモを書く
5. ...

作業開始！

●時▲分まで！

1. 公式サイトを開く
2. 作業メモを書く
3. 概要を読む
4. 作業メモを書く
5. ...

ふりかえり1

…●時▲分！

- ✓ これまでの活動を見直す機会
- ✓ 目的：
 - ✓ 他の人の視点での考え方を知る
 - ✓ 作業ログが役に立つことを実感

ふりかえり1：デモ デモ

- ✓ ビギナー：
 - ✓ 作業メモを読む
- ✓ サポーター：
 - ✓ 気になることをビギナーに質問
 - ✓ フィードバックポイントを確認
 - ✓ 完了→issueにコメント

ふりかえり1：進め方

- ✓ サポーターを他の人に交代
- ✓ 対象ビギナーの作業ログをディスプレイに映す
- ✓ ビギナーが作業メモを読む
- ✓ 時間が余ったら：
 - ✓ 近くの他のビギナーにも説明

休憩

●時▲分まで！

現状確認

1. ~~ユーザーとして動かす~~
2. ~~ふりかえり1~~
3. ~~フィードバックポイント~~を
発見!
4. ↑を**フィードバック**

フィードバック



- ✓ upstream（開発元）に
うまくいかなかったことを報告
 - ✓ ここで詰まった、を伝える
 - ✓ こうだったらよかった、を伝える

報告方法

1. 整理する
 - ✓ 自分の考えが文章になればOK
2. **開発者にとって**
わかりやすくなるように編集
3. 適切な場所に報告
 - ✓ GitHubのissueとか

1. 整理する

- ✓ 自分で自分の気持ちを理解
 - ✓ 自分が読んで理解できる文章にまとめられれば理解できている
 - ✓ 自分が理解できていないことは開発者にも伝えられない！
 - ✓ 作業メモに追記→サポーター確認

サポーターへ：メモ（断片）の文書化を手伝って
例：考えを整理できるような質問をする

整理方法

デモ

- ✓ 作業メモを開く
- ✓ フィードバック対象を決める
- ✓ 自分の気持ちを作業メモに追記
- ✓ サポーターに確認依頼

2. 編集する

- ✓ **開発者にとって**
わかりやすくなるように編集
- ✓ 報告方針をまとめているOSSもある
例：GitHubにあるCONTRIBUTING.md
- ✓ 作業メモに追記→サポーターに確認

サポーターへ：リーダブル化を手伝って
例：自分が開発者ならこう読めると開発者視点を伝える

編集の仕方



- ✓ ポイント
 - ✓ **相手**がわかるように書く
 - ✓ 例：省略しない（具体的に書く）

省略例

“インストールしました。
動きませんでした。
どうしたらいいのでしょうか？”

”

省略しない例

“
↓でインストール

```
$ sudo apt install ...  
(...実行結果...)
```

↑のように失敗しました。
環境：Ubuntu 24.04 amd64

”

なぜ省略しないか

- ✓ 相手は私を知らないから
 - ✓ 省略すると**想像**しないといけない
 - ✓ だいたい**想像は外れる**
 - ✓ 話が噛み合わない！

省略しないとは

- ✓ 詳細を書く
 - ✓ 実行したコマンド・実行結果
- ✓ やったことを書く
- ✓ やっていないことを書く
- ✓ 期待した結果を書く

編集方法



デモ

- ✓ 作業メモを開く
- ✓ 自分の気持ちを開発者に伝わるように
まとめて作業メモに追記
- ✓ サポーターに確認依頼

3. 報告する

- ✓ 適切な場所に報告
 - ✓ OSSによって報告場所は違う
- ✓ サポーターへ
 - ✓ 報告に二の足を踏んでいる人の背中を押してあげて
例：開発者視点を伝える：自分ならこの報告をもらったら助かる

報告方法



デモ

- ✓ 報告方法を探す
- ✓ サポーターに後押ししてもらう
- ✓ まとめた報告内容を報告

報告

●時▲分まで！

1. 整理する
2. 開発者にとってわかりやすくなるように編集する
3. 適切な場所に報告する
4. 報告したことをチャットで自慢！

✓ <https://app.element.io/#/room/#oss-gate:matrix.org>

ふりかえり2：デモ デモ

- ✓ ビギナー：
 - ✓ 作業メモを読む
- ✓ サポーター：
 - ✓ **よかったことをよい！**という
 - ✓ 気になることをビギナーに質問
 - ✓ 完了→issueにコメント

ふりかえり2：進め方

- ✓ サポーターを他の人に交代
- ✓ 対象ビギナーの作業ログをディスプレイに映す
- ✓ ビギナーが作業メモを読む
- ✓ 時間が余ったら：
 - ✓ 近くの他のビギナーにも説明

OSS開発関連情報



✓ 生成AI

生成AIとOSS開発

- ✓ OSS開発にも生成AIは使われ始めている
- ✓ よい付き合い方はまだ探し探し
- ✓ これは困るというのは見えてきている
- ✓ ポイント：
自分も開発チームの1人という振る舞い

生成AIで報告

- ✓ 生成された報告は冗長なことが多い
 - ✓ 無用な冗長さは開発者を疲弊させる
 - ✓ 開発チームの1人としてどうしたい？
- ✓ 必須：報告前に報告者がレビュー！
 - ✓ 報告者がそもそも理解できていること
 - ✓ 開発者に聞かれたら自分で答えられるくらい
 - ✓ 報告者が不要な情報を減らす

生成AIでプルリクエスト



- ✓ 大きなプルリクエストになりがち
 - ✓ 大きなプルリクエストは開発者を疲弊させる
 - ✓ 開発チームの1人としてどうしたい？
- ✓ 必須：作成前に作成者がレビュー！
 - ✓ 詳細を理解する
 - ✓ レビュー可能なサイズにする

プルリクエストとレビュー



- ✓ 狙いの1つ：関係者間での知識の共有
 - ✓ 作成者が未理解→知識の共有は進まない
 - ✓ 作成者抜きで開発者が直接生成AIを使えばよい
- ✓ 開発チームの1人としてどうしたい？
 - ✓ 学んで次のプルリクエストに活かせる？
 - ✓ 学んだことを他の人に伝えられる？

生成AIとライセンス

- ✓ 機械が生成したものに著作権は発生しない
- ✓ 著作権があるコードの複製を生成しうる
 - ✓ そのコードのライセンスが大事
 - ✓ ライセンスを守れば再利用可能
- ✓ ライセンスの問題がないことはプルリクエスト作成者が保証する

生成AIとコミット



- ✓ どのツールを使ったかの情報を残す
 - ✓ GitならGenerated-By: ...などで記録
 - ✓ なにか問題があったときに調べられるように

生成AIとOSS開発



- ✓ 生成したものを丸投げはダメ
- ✓ 生成したものを詳細まで理解した上で活用

まとめ

- ✓ 今日やったことを再確認
- ✓ 明日からのことを確認

目的の確認

OSS開発**未**経験者
↓
OSS開発 経験者

やったこと



OSS開発参加を体験する

1. ユーザーとして動かす
2. フィードバック

体験時のポイント



常にメモ

常にメモの理由



- ✓ 詰まったところに気づくため
 - ✓ いつもはスルーしていない？
 - ✓ 実はフィードバックポイント！

詰まったところ



- ✓ OSS開発参加の**チャンス**！
 - ✓ ポジティブに捉えてみよう
 - ✓ 実際に参加して楽しかった？
- ✓ 直ると次の人は**うまいく**
 - ✓ 気分がいいね！

気づいた？

- ✓ コードを書くだけがOSS開発参加方法じゃない
 - ✓ 使いはじめてのユーザーだからできることもある
- ✓ やり方を知ればやれる
 - ✓ 明日からもやってみよう！

明日からのやり方



- ✓ 自分が使っているOSSでもやってみよう
 - a. ユーザーとして動かす
 - b. 気になったことをまとめる
 - c. フィードバック
- ✓ ↑失敗が怖い？

OSSと失敗

- ✓ そもそも失敗と認識されない
 - ✓ 少なくとも1発アウト！はほぼない
 - ✓ 新規開発者は基本的にWelcome
- ✓ 失敗しても根に持たれない
 - ✓ 失敗→改善：改善後を評価



明日からオススメ方法をTry！

メッセージ

不安がらずに
OSSの開発を
楽しんで！

来てよかった！と思ったら



- ✓ OSS Gateに継続参加！
 - ✓ チャット・ワークショップ・…
 - ✓ 社内・コミュニティ向けワークショップを検討
- ✓ OSS Gateを継続宣伝！
 - ✓ 周りの人を誘う
 - ✓ ブログに書く

OSS Gateはメンバー募集中



- ✓ 次のワークショップ開催日は…
 - ✓ 開催する毎に門をくぐる人がいる！
 - ✓ 多くの人と協力して継続したい！
 - ✓ 会場提供・サポーター・進行役・…
- ✓ 次回を開催日をチェック！
 - ✓ <https://app.element.io/#/room/#oss-gate:matrix.org>
 - ✓ アナウンス：oss-gate/announce

おねがい

- ✓ 今日のフィードバックを！
 - ✓ 次に活かしたい
- ✓ この後すぐ
 - ✓ アンケート記入
 - ✓ アンケート結果をみんなで確認

アンケートの回答方法



1. `github.com/oss-gate/workshop` をfork
2. `tutorial/retrospectives/YYYY-MM-DD-***`
 - ✓ `cp beginner.yaml beginner-***.yaml`
 - ✓ `cp supporter.yaml supporter-***.yaml`
3. `git add` → `git commit` → `git push`
4. `github.com/自分のアカウント/workshop`を開いて「Pull request」