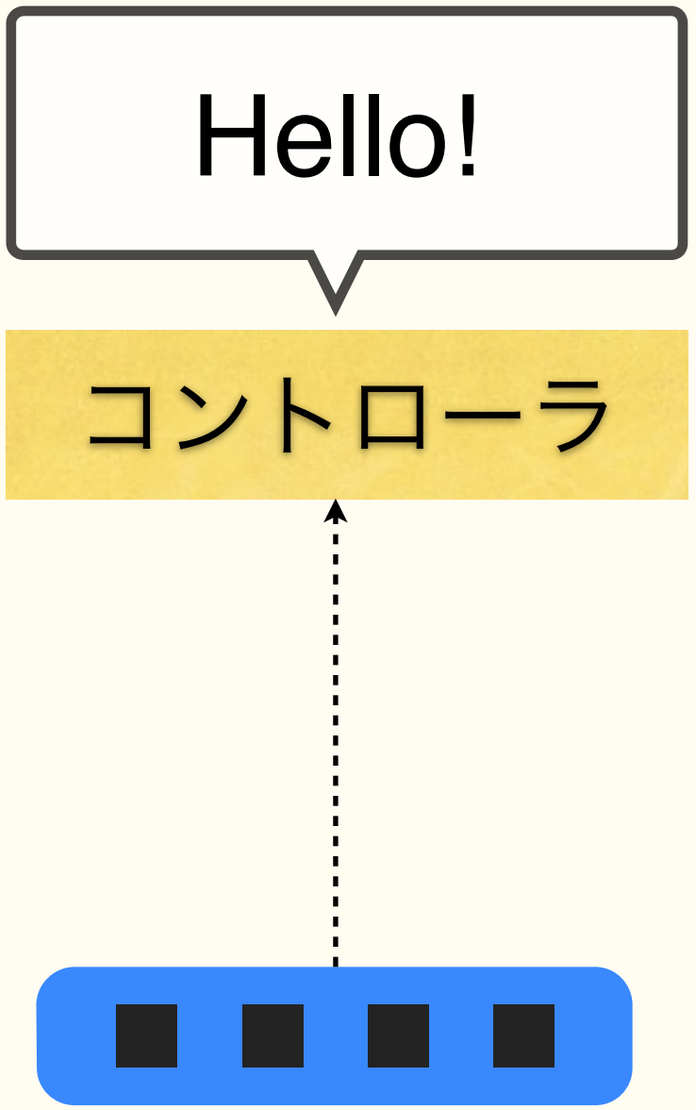
Hello, Trema!

高宮 安仁 @yasuhito



仮想NW

ルーティング
スイッチ

トポロジ
ディスカバリ

スイッチ

Hello
Trema

課題用リポジトリ

- ユーザ名/hello_tremaリポジトリを作ろう
- https://github.com/handai-trema/syllabus で、
  - 課題 → hello_trema をクリック
  - 右上の Fork ボタンをクリック

$ git clone [リポジトリのURL]

$ cd リポジトリのディレクトリ

$ bundle install —binstubs

- bundleコマンドでtremaなど必要なモノを./bin/へインストール

```
$ ./bin/trema run ./lib/hello_trema.rb
Trema started.
```

- `trema run` [Rubyスクリプト]
- `Ctrl-C` で停止

```
Trema started.
```

```
trema run
```

起動

hello_trema.rb
(コントローラ)

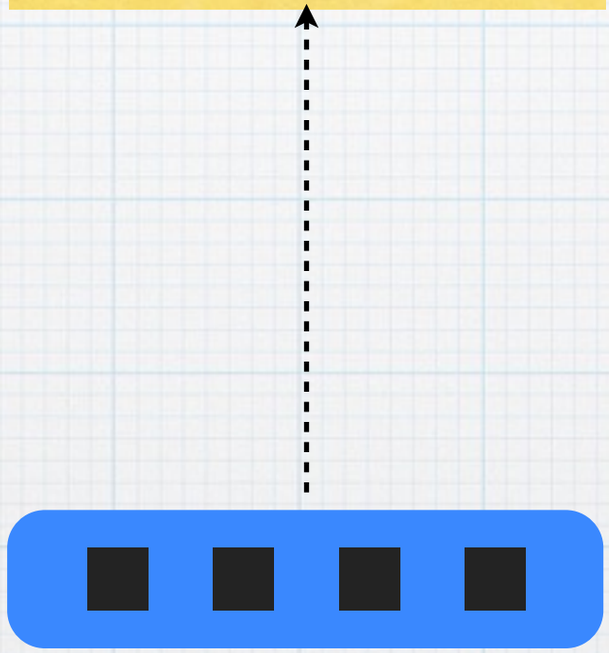
```
$ ./bin/trema run ./lib/hello_trema.rb \
  -c trema.conf

Trema started
Hello 0xabc!
```

- -cオプションで仮想スイッチをコントローラに接続

Hello 0xabc!

コントローラ



dpid = 0xabc

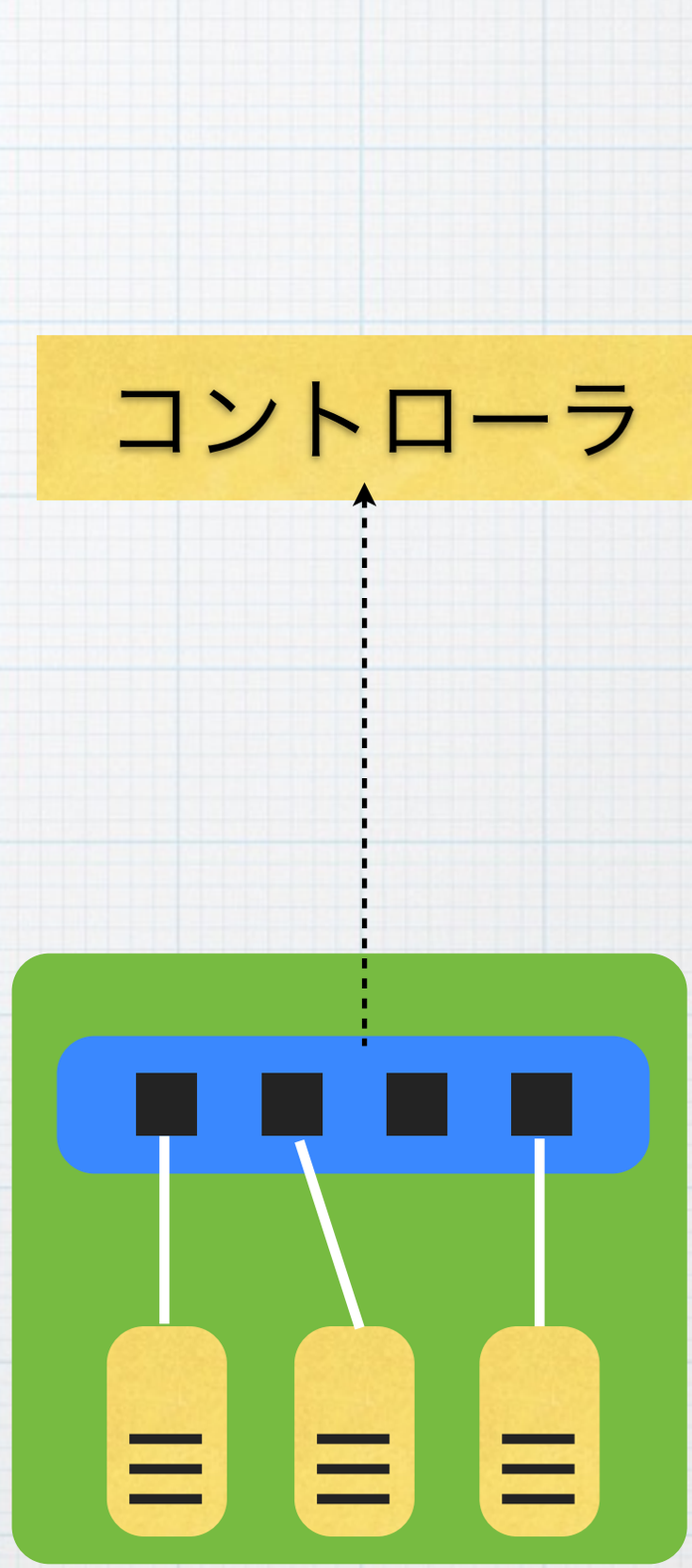
trema run
-c file

起動

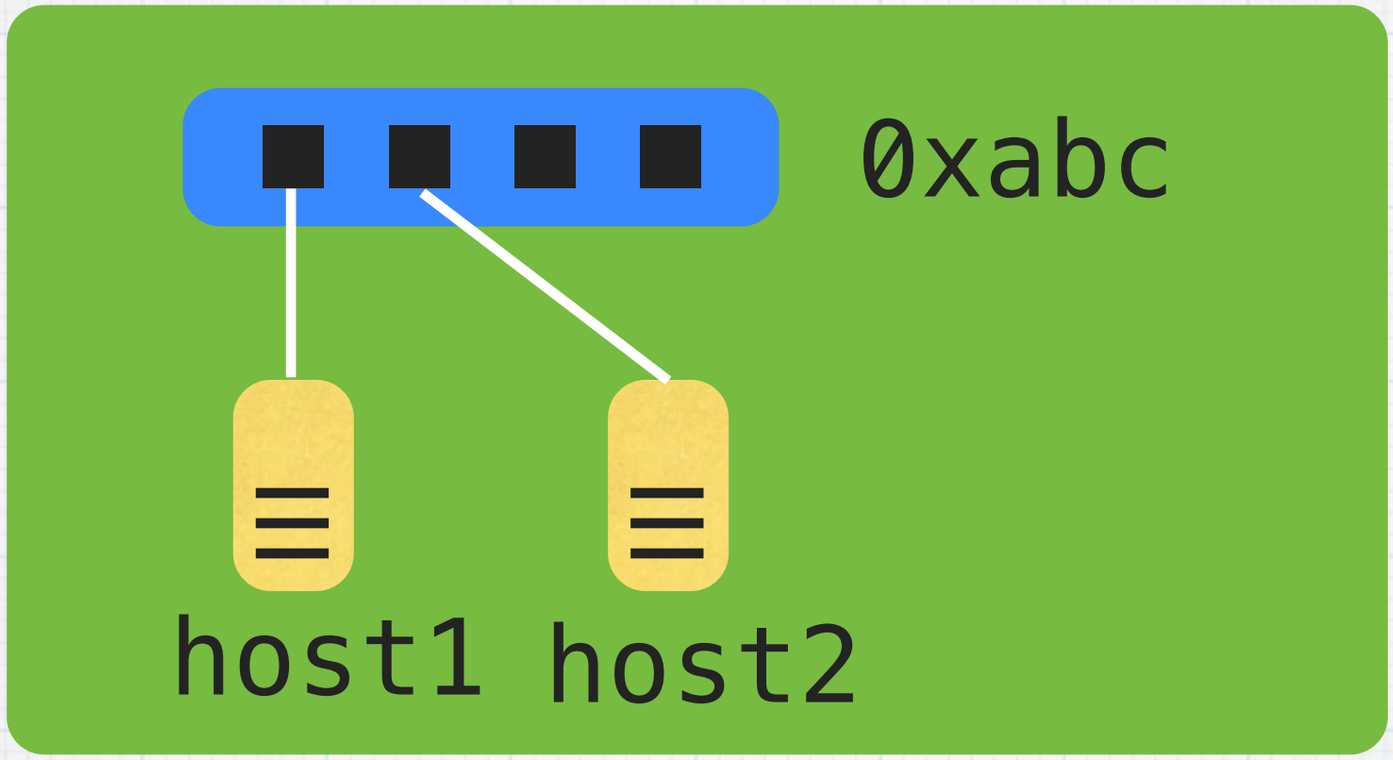
コントローラ

作成

ネットワーク
DSL



仮想ネットワーク




```
vswitch { dpid 0xabc }
vhost(‘host1’)
vhost(‘host2’)
link ‘0xabc’, ‘host1’
link ‘0xabc’, ‘host2’
```

仮想
ネットワーク

テスト

trema コマンド

実行

デバッグ

API
リファレンス

```
trema run
trema help
trema killall
trema version
trema ruby

trema stop
trema start
trema send_packets
trema show_stats
trema dump_flows
trema netns
```

「Rubyの書き方は?」

品詞+文法

登場する品詞

- キーワード
- 名詞
- 動詞

キーワード(予約語)

```
alias and BEGIN begin break case
class def defined do else elsif
END end ensure false for if in
module next nil not or redo
rescue retry return self super
then true undef unless until
when while yield
```

```
class HelloTrema < Trema::Controller
  def start(_args)
    logger.info ‘Trema started.’
  end
end
```

・キーワードは構造を決める

```
class HelloTrema < Trema::Controller
  def start(_args)
    logger.info ‘Trema started.’
  end
end
```

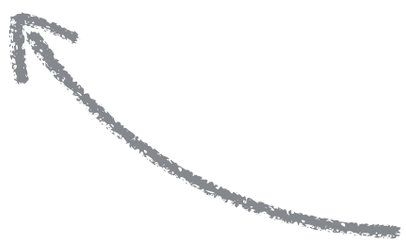
- クラス定義 (class … end)
- 子クラス < 親クラス名

固有名詞=定数

Tsutenkaku

Rainbow Bridge

HelloTrema



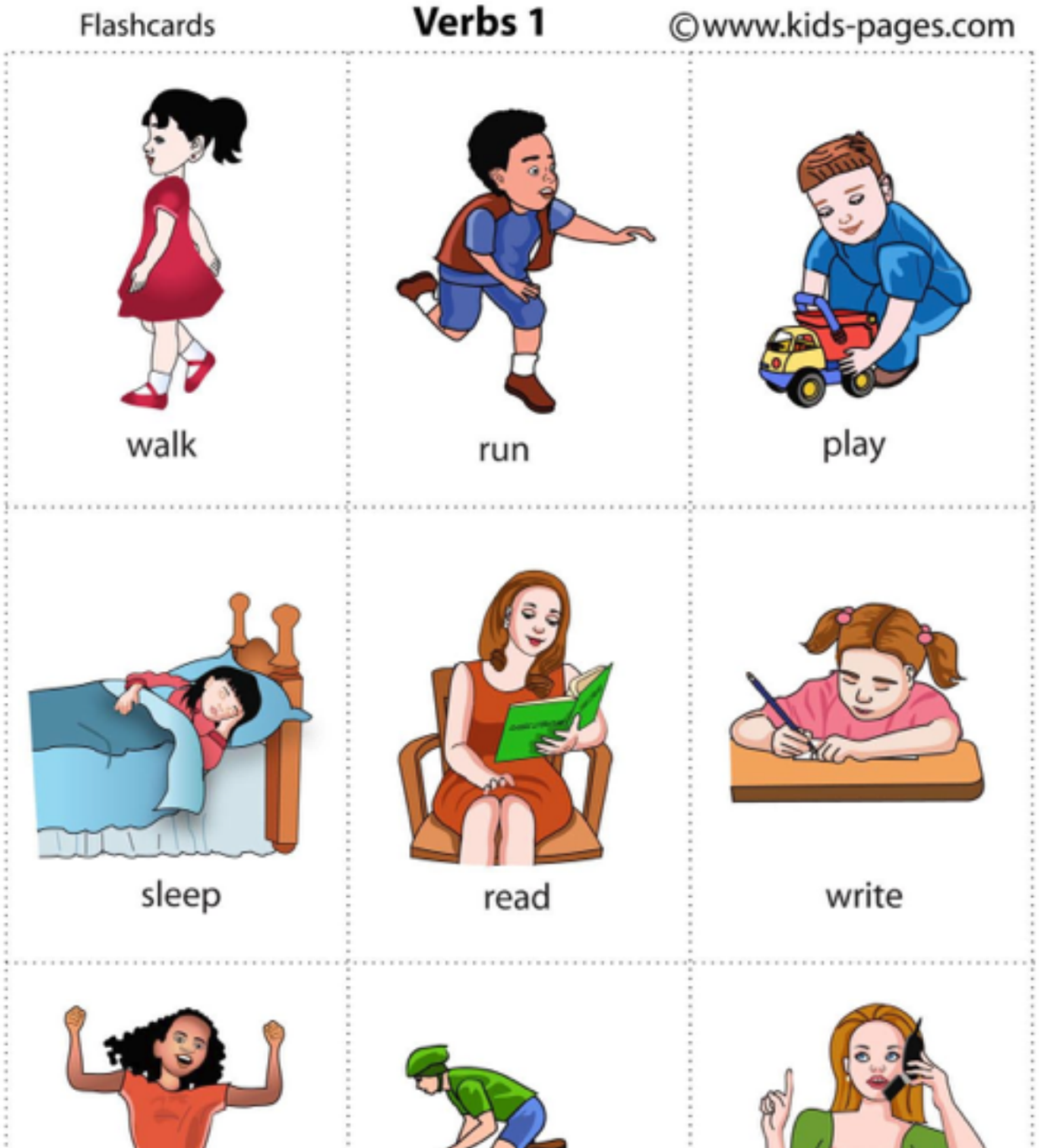
- 大文字で始まる
- 内容を変更できない

```
class HelloTrema < Trema::Controller
  def start(_args)
    logger.info ‘Trema started.’
  end
end
```

- 定数は大文字で始まる

メソッド=動詞

- boy.run
- girl.write ‘abc’
- logger.info

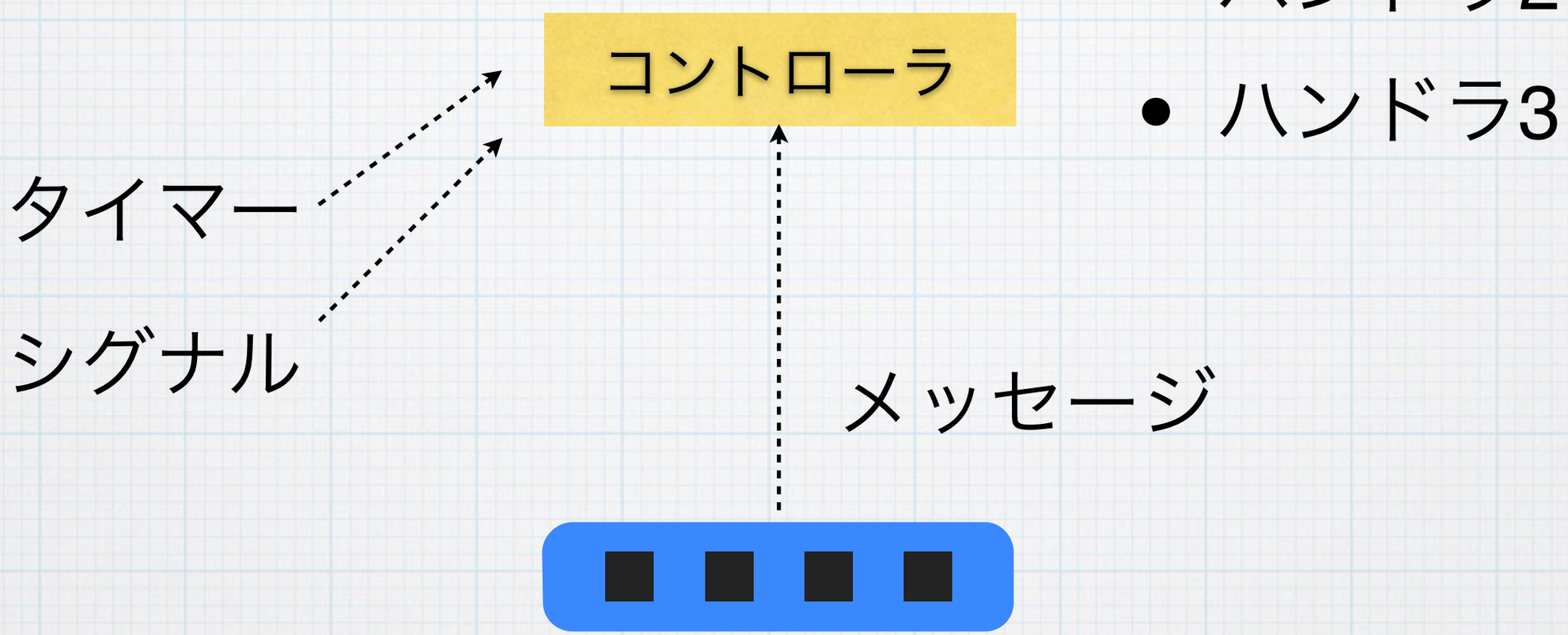


```
class HelloTrema < Trema::Controller
  def start(_args)
    logger.info ‘Trema started.’
  end
```

- logger = ロガー
- info = info ログの出力

```
class HelloTrema < Trema::Controller
  def start(_args)
    logger.info ‘Trema started.’
  end
end
```

- def … end = メソッド定義
- start はハンドラメソッド



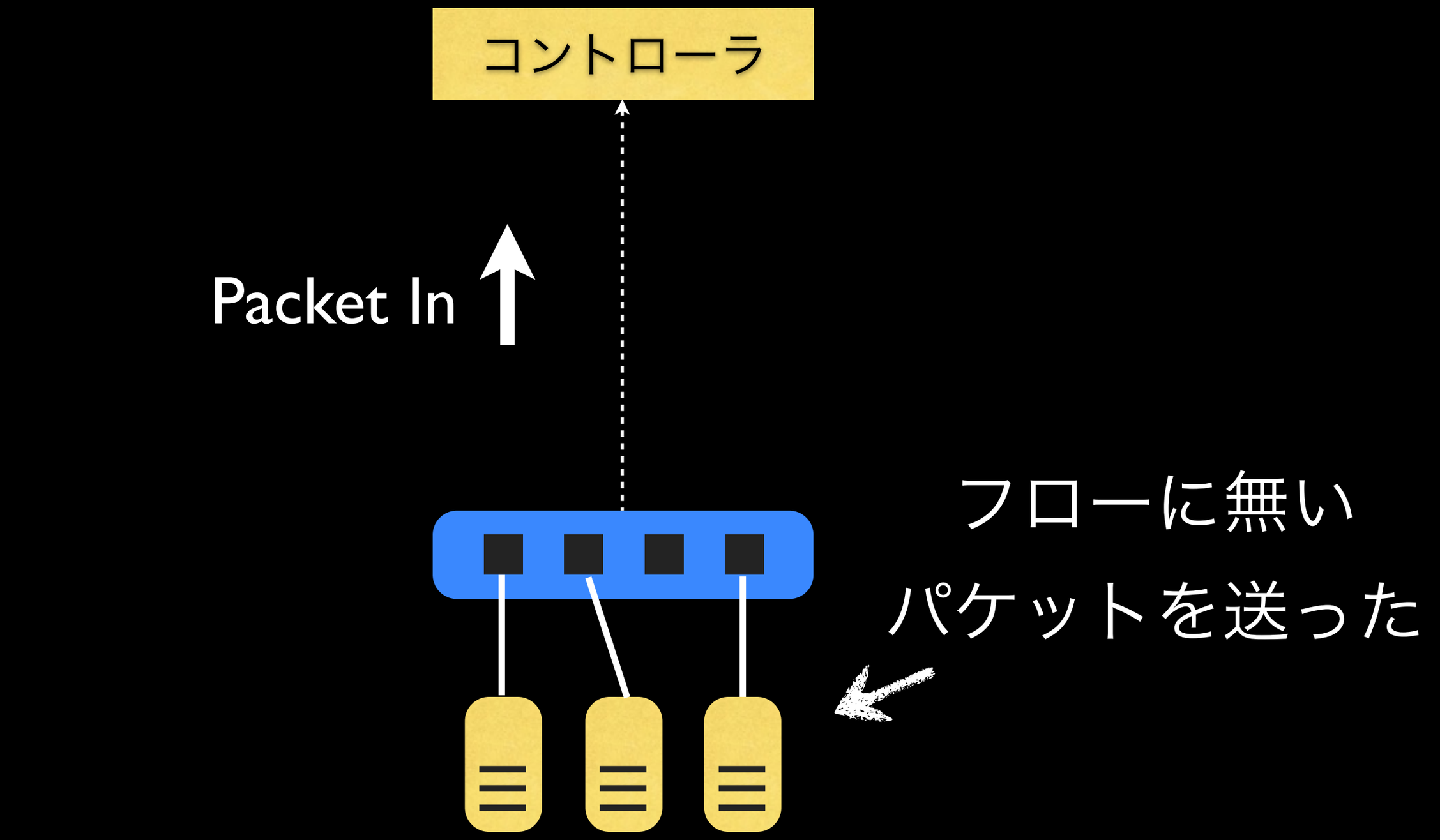
コントローラの外界からのメッセージや
イベントに応じ、コントローラの対応する
ハンドラメソッドが呼ばれる

start()

コントローラが起動した

コントローラ

packet_in()



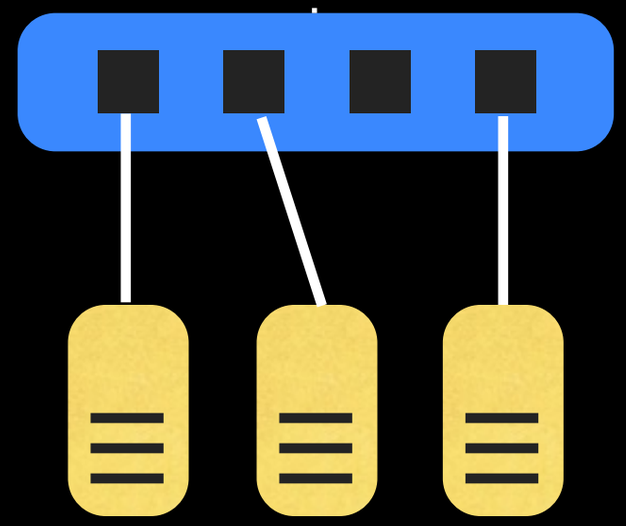
flow_removed()

コントローラ

Flow Removed

フローが

expire した



ハンドラー覧

start switch_ready
switch_disconnected
packet_in flow_removed
port_status openflow_error
features_reply stats_reply
barrier_reply
get_config_reply
queue_get_config_reply
vendor

課題

- スイッチを停止したら

“Bye 0xabc”

と表示せよ ← スイッチのDPID

- 接続を捕捉
- 切断を捕捉

```
class FooBar < Trema::Controller
  def switch_ready(dpid)
    # ...
  end

  def switch_disconnected(dpid)
    # ...
  end
end
```

```
% trema run foobar.rb -c network.conf
```

```
vswitch { dpid 0x1 }

vswitch { dpid 0x2 }

vswitch { dpid 0x3 }
```

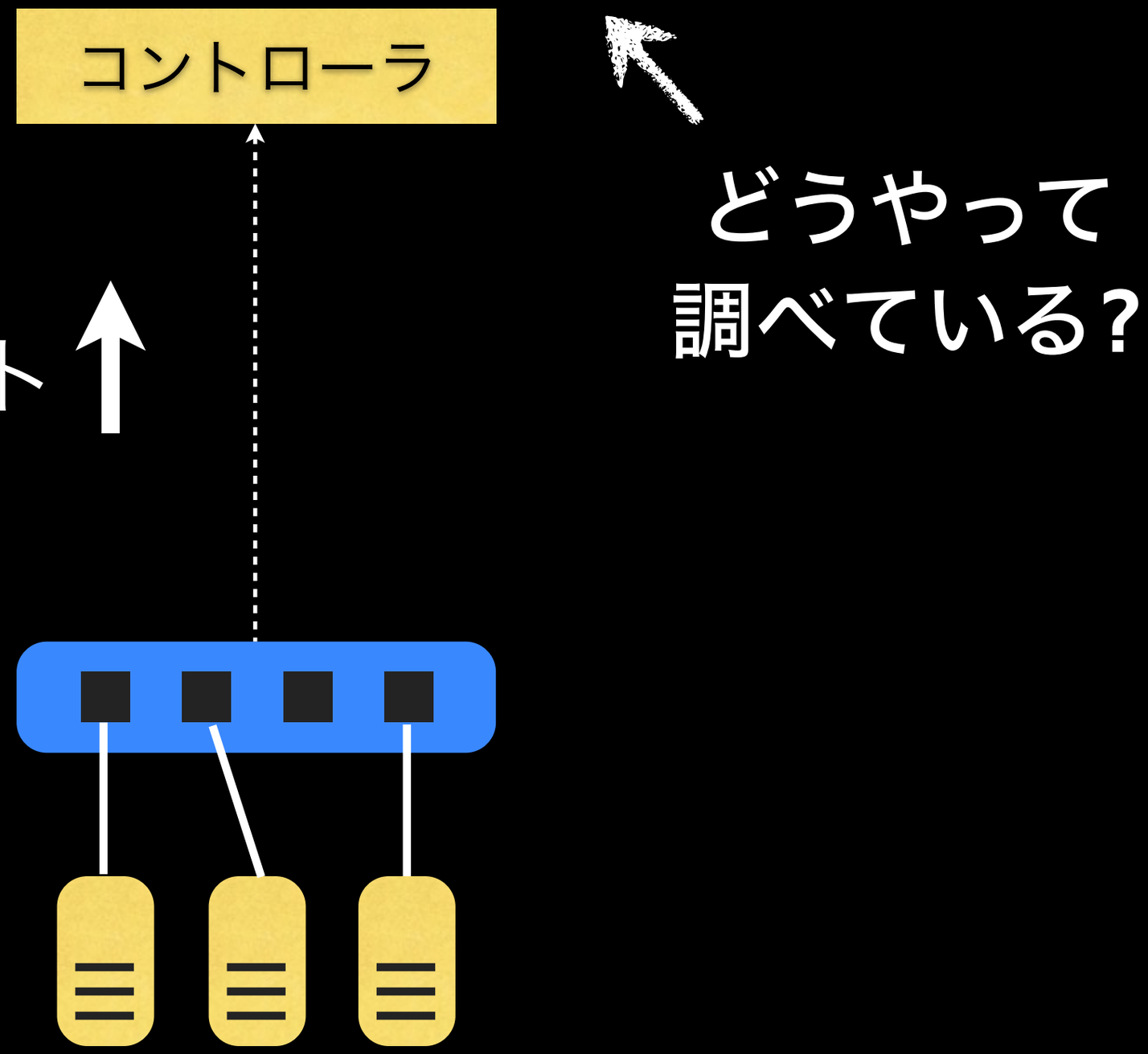
```
% trema stop 0x1   # 仮想スイッチを殺す

% trema start 0x1   # 仮想スイッチを復活
```

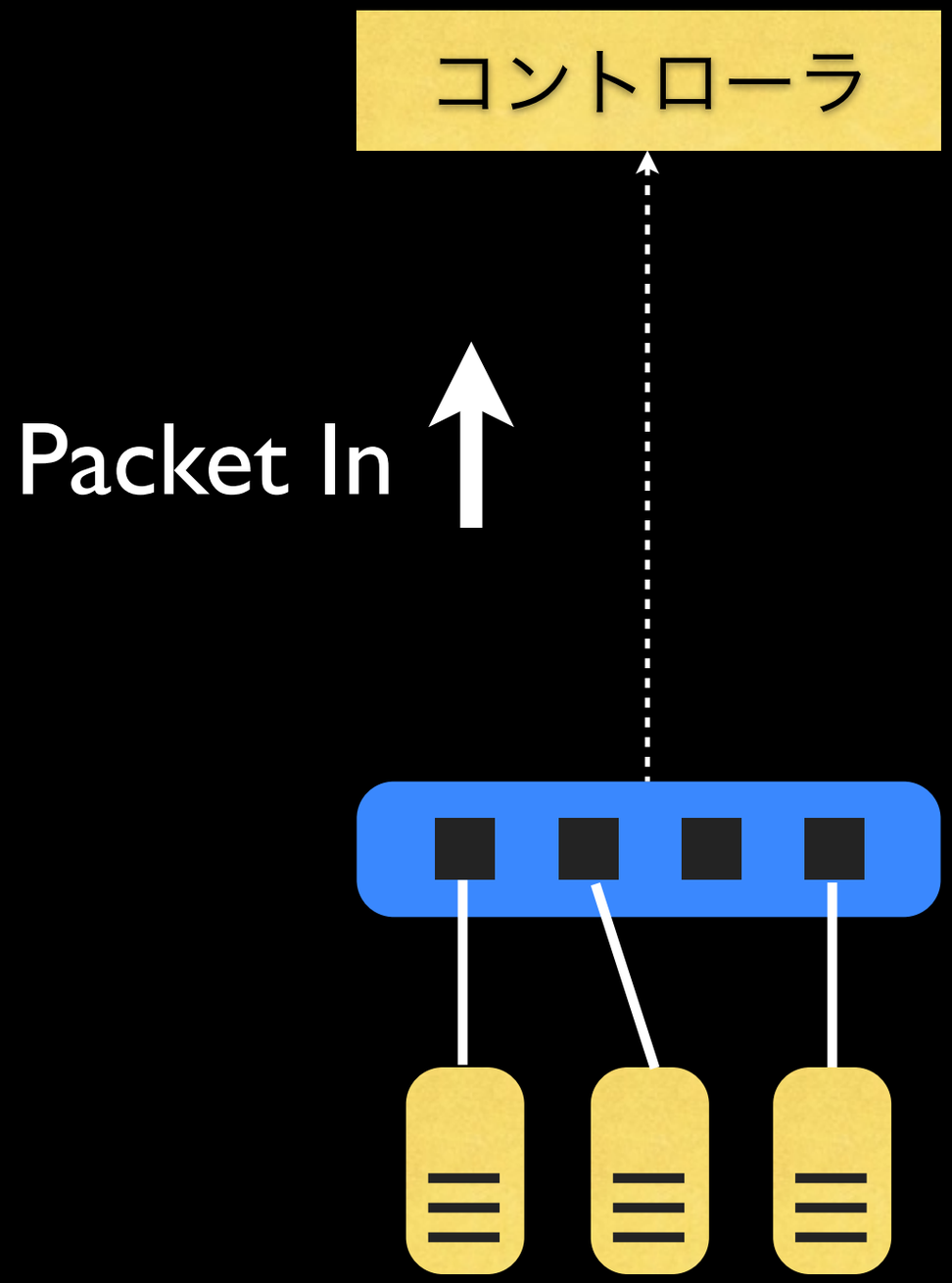
- trema stop/start コマンドで仮想スイッチを操作

「なぜこれだけで動く？」

ハンドラがある？ ない?



Controllerクラスの仕事



1.自分が packet_in()
を持っているか?

2.あれば呼ぶ

3.なければ呼ばない

リフレクション
(自己反映計算)

実行時 (!=コンパイル時) に自分を知る

- メソッドの一覧
- クラス名

うまく使えば、コードを短く頑健にできる

レポート課題 1

- HelloTrema を改造して、
  “Hi! from HelloTrema”
  と表示せよ

クラス名

```
class HelloTrema < Trema::Controller
  def start(_args)
    logger.info ‘Hi! from HelloTrema’
  end
end
```

- イマイチな解答例
- `HelloTrema’が2箇所にある

DRY
(Don’t Repeat Yourself)

- コピペをなくし、変更に強いコードを
- Ruby っぽい書き方 (イディオム) を覚えよう

```
class HelloTrema < Trema::Controller
  def start(_args)
    logger.info ‘Trema started.’
  end
end
```

・インデントはスペース2個

・使わない引数は`_`で始める

Rubyっぽい書き方

- Ruby Style Guide
  https://github.com/bbatsov/ruby-style-guide
  - 日本語版もあります
- スタイル自動チェックツール
  - rubocop
  - bundle exec rake rubocop で自動チェック

レポートの出しかた

1. ユーザ名/hello_trema にファイルを追加
    - lib/hello_trema.rb を改造
    - report.md: コードの説明などを書く
    - 学籍番号や名前は書かなくていいです!
2. self_intro にリポジトリへのリンクを追加