








# Code is 量

ジーズアカデミー福岡主任講師 大杉太郎

# 自己紹介

- 氏名: 大杉太郎
- Twitter: @taroosg
- 仕事: エンジニア, プログラミング講師
- 技術: Laravel, JS, Deno, (Rust)
- 好きなもの:       
- 入学おめでとうございますッ!!!

# はじめに（一番言いたいこと）

学ぶことが目的ではない。

- アイデアを実現するプロダクトをつくる
- プロダクトをつくれる人間になる

つくれるようになるにはつくるしかない。

つまり、「Code is 量」だッ！

# Why Code is 量？

✗ 質

✓ 量

# もくじ

- 量はなぜ必要か
- 量をどうやって作るか
- まとめ

量はなぜ必要か

# 量はなぜ必要か

人間の「理解」の構造が原因

- 初めての人「コードが理解できない！！」

↑ なんで？？

# 理解 is 何?

理解 = 多くの具体例から法則を見出すこと (☑ 試行錯誤)

具体例 = 「書いたコード」と「動作結果」

∴ わからないときは「具体例」が足りない!



# 具体例を増やす方法

## たくさん試行錯誤する

- 材料をたくさん揃える.
- 「試行回数」が具体例の「量」「多様性」を生み出す.
- わからなくても手を動かしてコードを書くことが大事!

# 量はなぜ必要か

- 理解するには多様な材料が必要.
- 多様な材料を集めるには量が必要.

プログラミングは「できた」 → 「わかった」の順番!

つまり, **Code is 量!**

量をどうやって作るか

量  $\propto$  時間

量を実現するには時間が必要！

# 時間をどうやって作るか

時間はない！

- 人間の 24 時間は埋まっている．
- すでに使われている時間を削る．

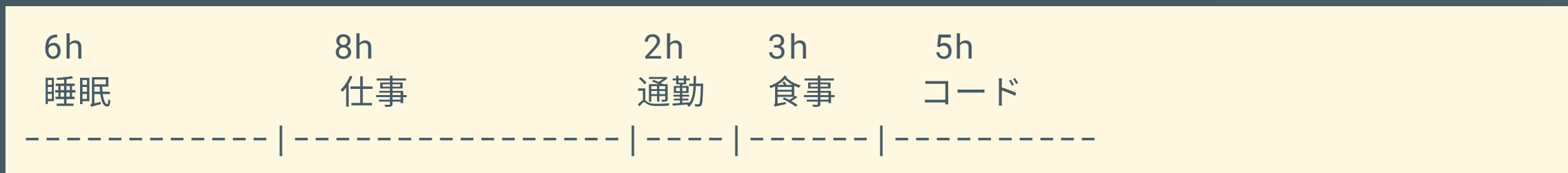
まず 1 日のスケジュールを書いてみましょう．

# 例 1



- 絶対に必要な時間とそれ以外を分ける.
- 「それ以外」を全部プログラミングにつぎ込む

# 例 1



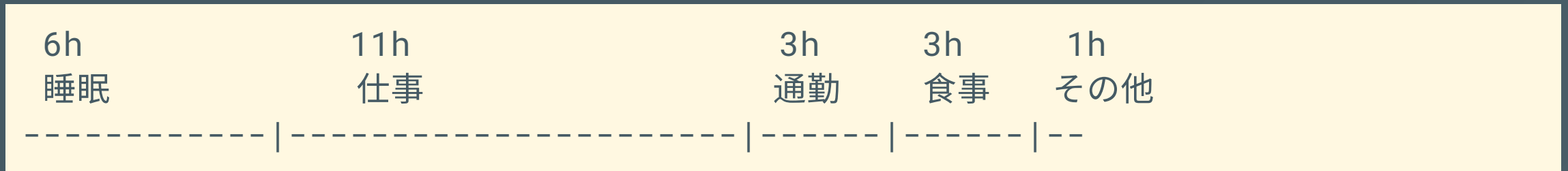
- Good

## Point

- 「優先順位」を決める！
- 「やらないこと」を明確にせよ！
- プログラミングは「最優先事項」！

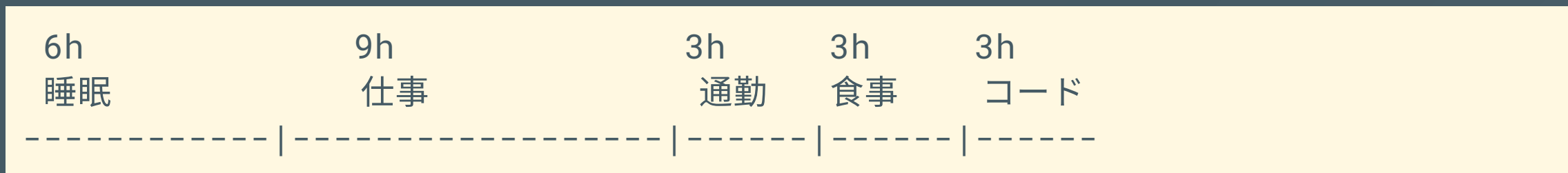


## 例 2



- 時短するしかない！

## 例 2



「質」を高めることで対応する。

- 必要なことをより短い時間で実現する。
- 仕事は経験値があるので時短できる。

# 時間をどうやって作るか

時間はないッ！ 今すぐ1日を可視化せよッ！

すでに使われている時間を削るしかない。

1. 優先順位（やらないこと）を決める。
2. 質を高めることで時間を作り出せる。

まとめ

# まとめ

## Code is 量!

- 「できる」 → 「わかる」の順番. 「できた」を増やすことが大事!
- 時間はない. やらないことを決めて質を高める!

**まずは時間を最大限確保しましょう！**  
**Code is 量！**



**Thanks!**

 alt text