

# テンプレートエンジンで HTMLを効率よく作成しよう

株式会社 FORK 高橋 学

# 自己紹介

---

高橋 学（たかはし まなぶ）

株式会社FORK 第1プロデュースユニット所属

フロントエンドエンジニア

2008年 FlashエンジニアとしてFork入社

最近是人材の育成やプロジェクトの管理・大規模案件の設計などを担当しています。

# FORKについて

---



渋谷にあるWeb制作会社 150名規模

LPI-Japan HTML5 アカデミック認定校

HTML5プロフェッショナル認定試験合格者

- ・レベル1 : 38名

- ・レベル2 : 5名

研究開発サイト

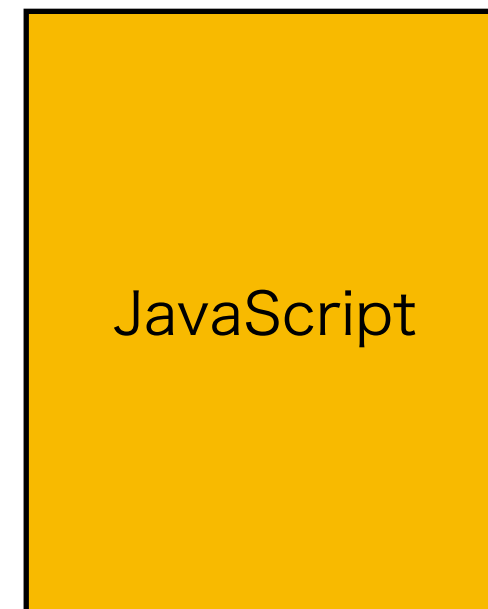
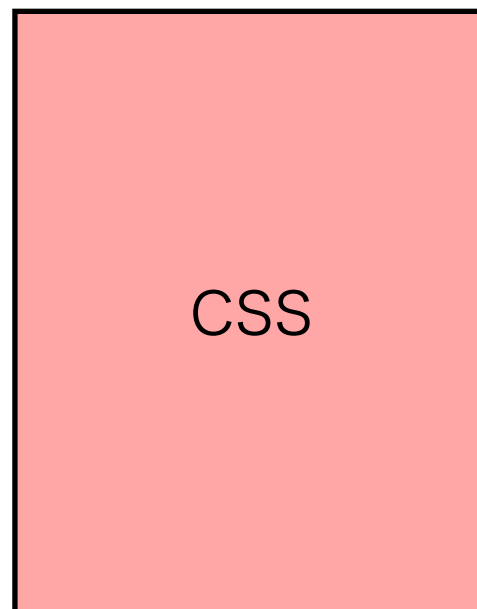
[4009.jp](http://4009.jp)

# 最近のフロントエンド

# 最近のフロントエンド

---

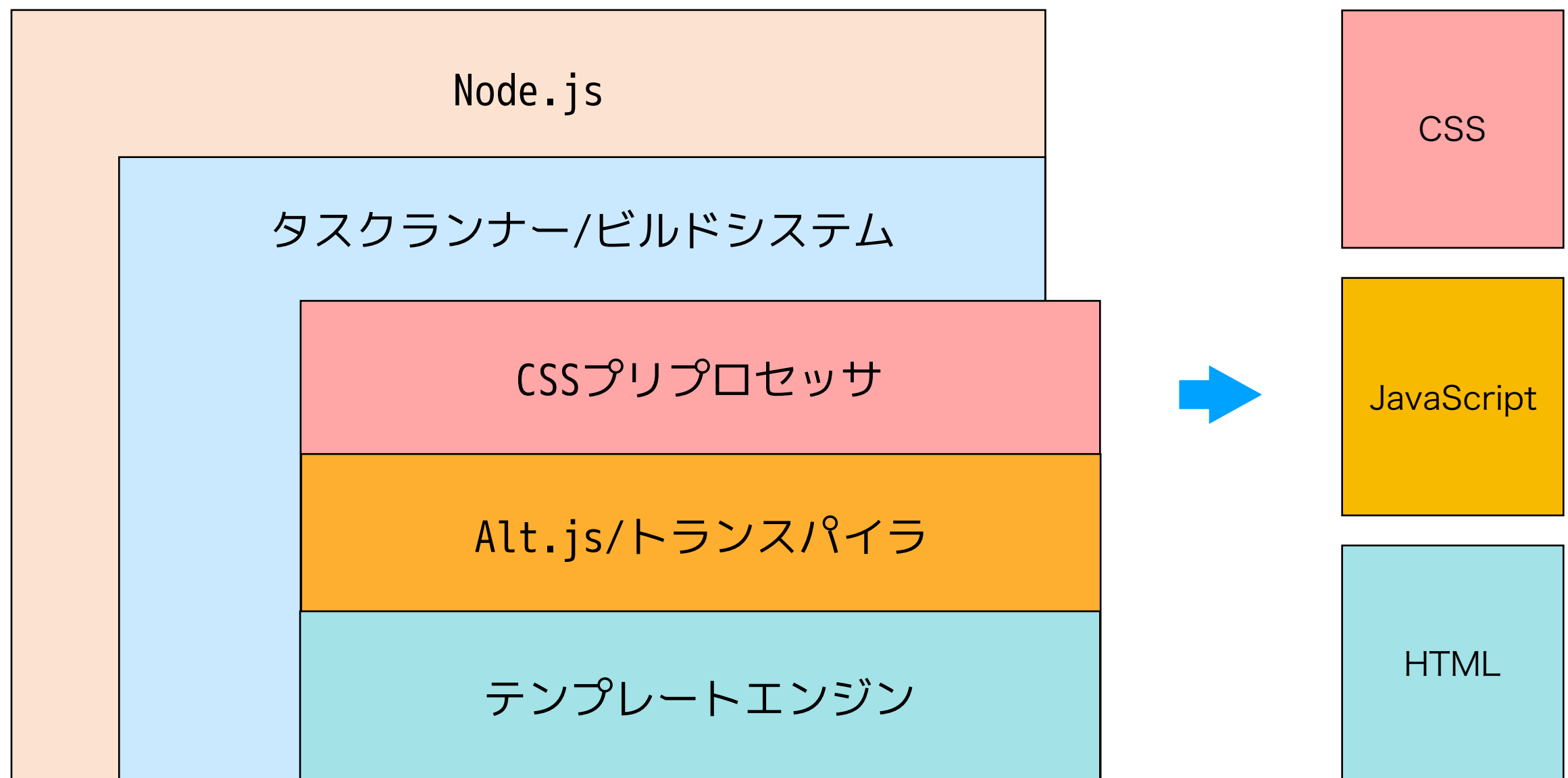
フロントエンドエンジニアのアウトプットは昔から変わりませんが、



# 最近のフロントエンド

---

それらは様々な方法で制作されています。



# テンプレートエンジン とは

# テンプレートエンジンとは

---

テンプレートファイルとデータから、HTMLを生成(表示)させる技術の総称。



# テンプレートエンジンについて

---

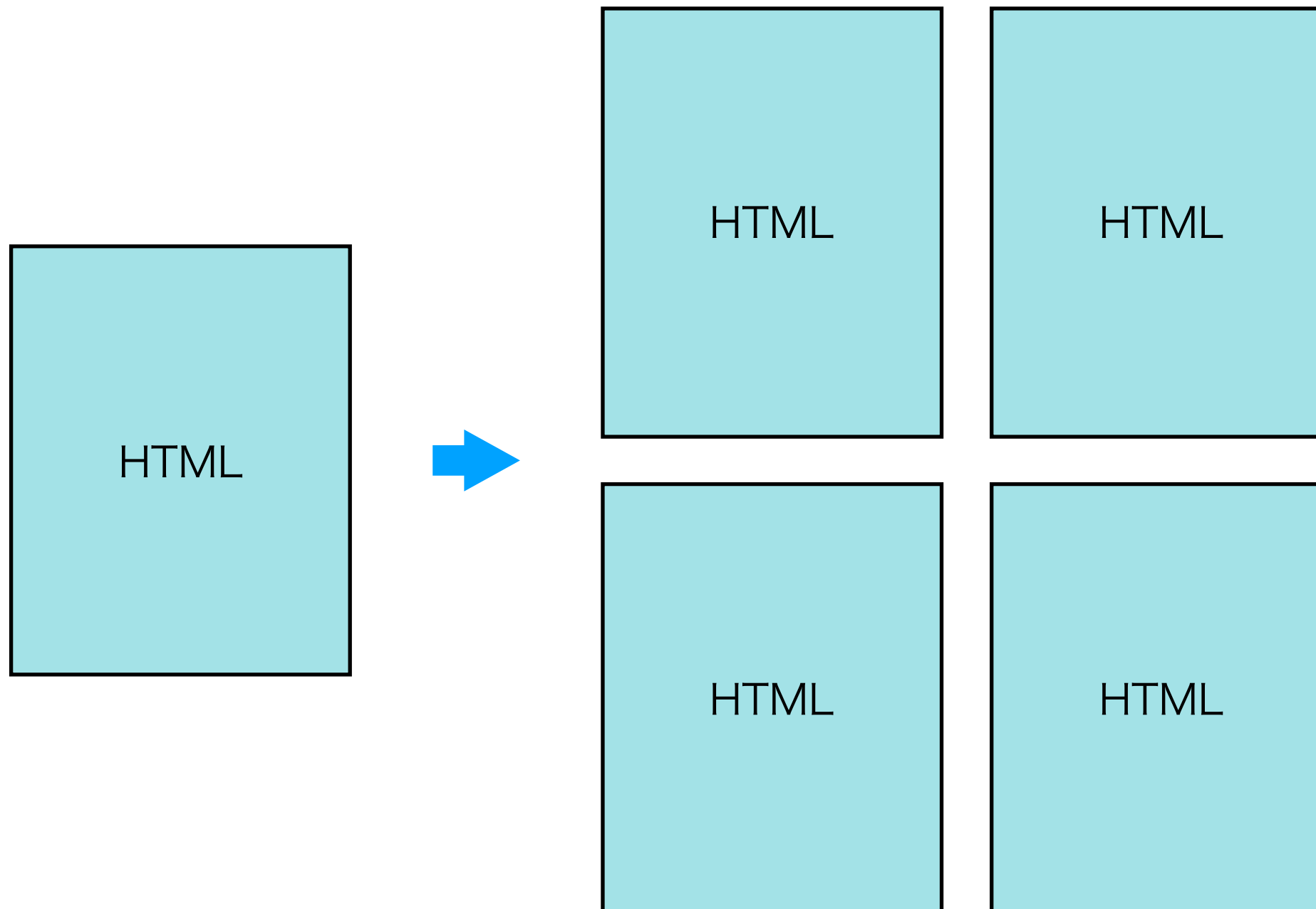
- **サーバーでの動的出力**
- **ブラウザでの動的出力**
- **静的HTMLファイルの出力**

※今回は静的HTMLファイルの出力に絞ってお話いたします。

# 旧来のHTMLの制作イメージ

---

ベースとなるhtmlを複製



# 旧来の問題

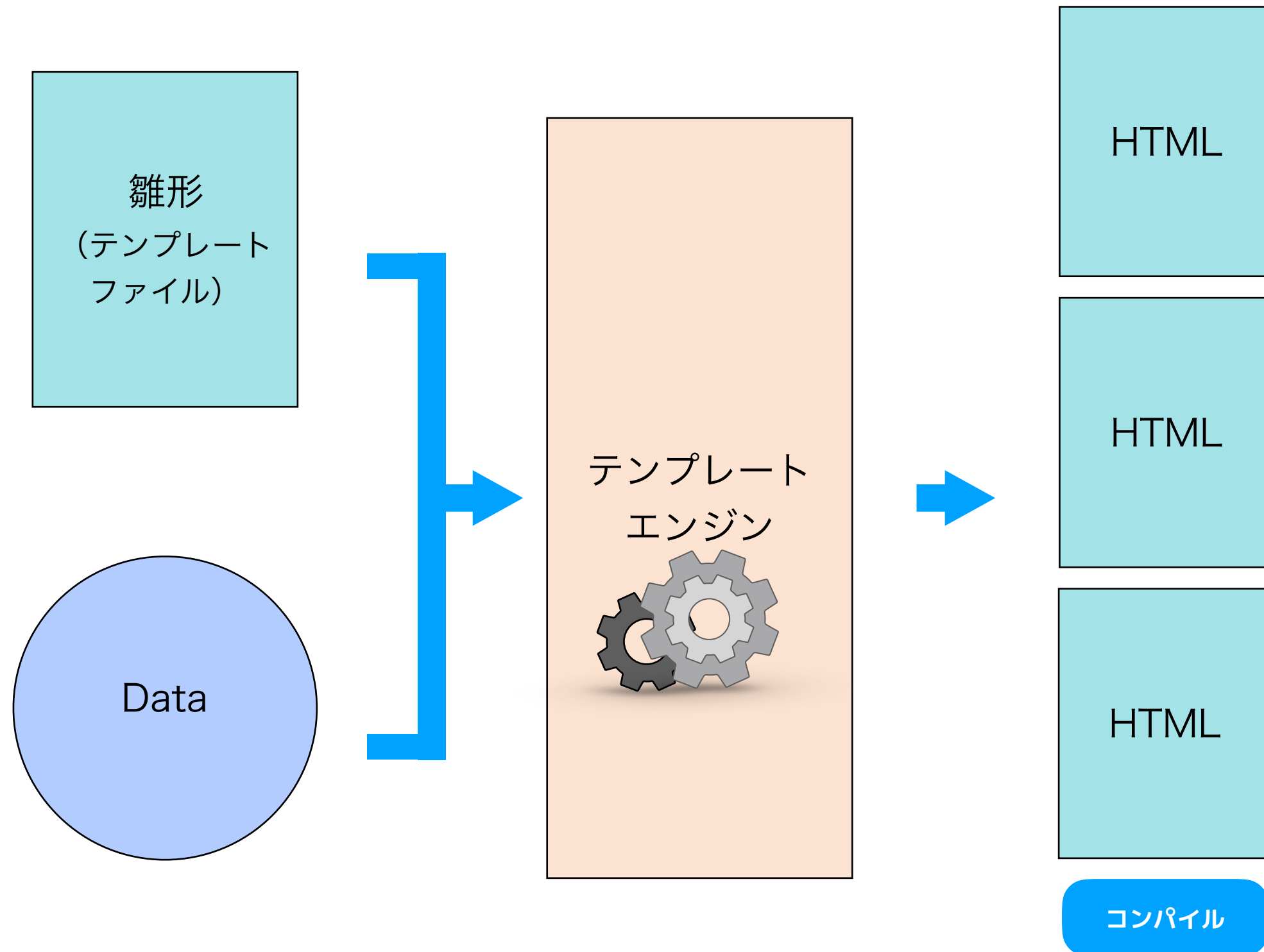
---

**レイアウトを修正 = ページ全てを修正**

**同じ書式で内容が違ふ部分はコーディングで対応**

**複数ページ修正時のミス（ヒューマンエラー）**

# テンプレートエンジンでの作成イメージ



# テンプレートエンジンのメリット

---

**変数・継承・インクルードなど便利機能が使える**

**共通部分を使いまわせる（パーツ化）**

**データをまとめて記述できる。（別ファイル化）**



**ミスを防げる/メンテナンスしやすい/作業効率UP**

# テンプレートエンジンとは

---

多くのテンプレートエンジンが存在し、それぞれ個性があります。メジャーなツールは下記になります。



# テンプレートエンジンとは

---

## htmlタグベースのテンプレートエンジン



# テンプレートエンジンとは

---

## 独自記法のテンプレートエンジン





# テンプレートエンジンとは

---

今回は「Pug」についてお話しします。



# テンプレートエンジンPug

主要なテンプレートエンジンのgithub Star 降順

pug	 Watch ▾	581	 Star	14,995	 Fork	1,741
handlebars.js	 Watch ▾	452	 Star	12,290	 Fork	1,609
mustache.js	 Watch ▾	412	 Star	11,060	 Fork	2,070
slim	 Watch ▾	584	 Star	8,238	 Fork	1,629
hogan.js	 Watch ▾	281	 Star	4,785	 Fork	401
haml	 Watch ▾	96	 Star	3,195	 Fork	519
ejs	 Watch ▾	64	 Star	1,653	 Fork	210

# Pugについて

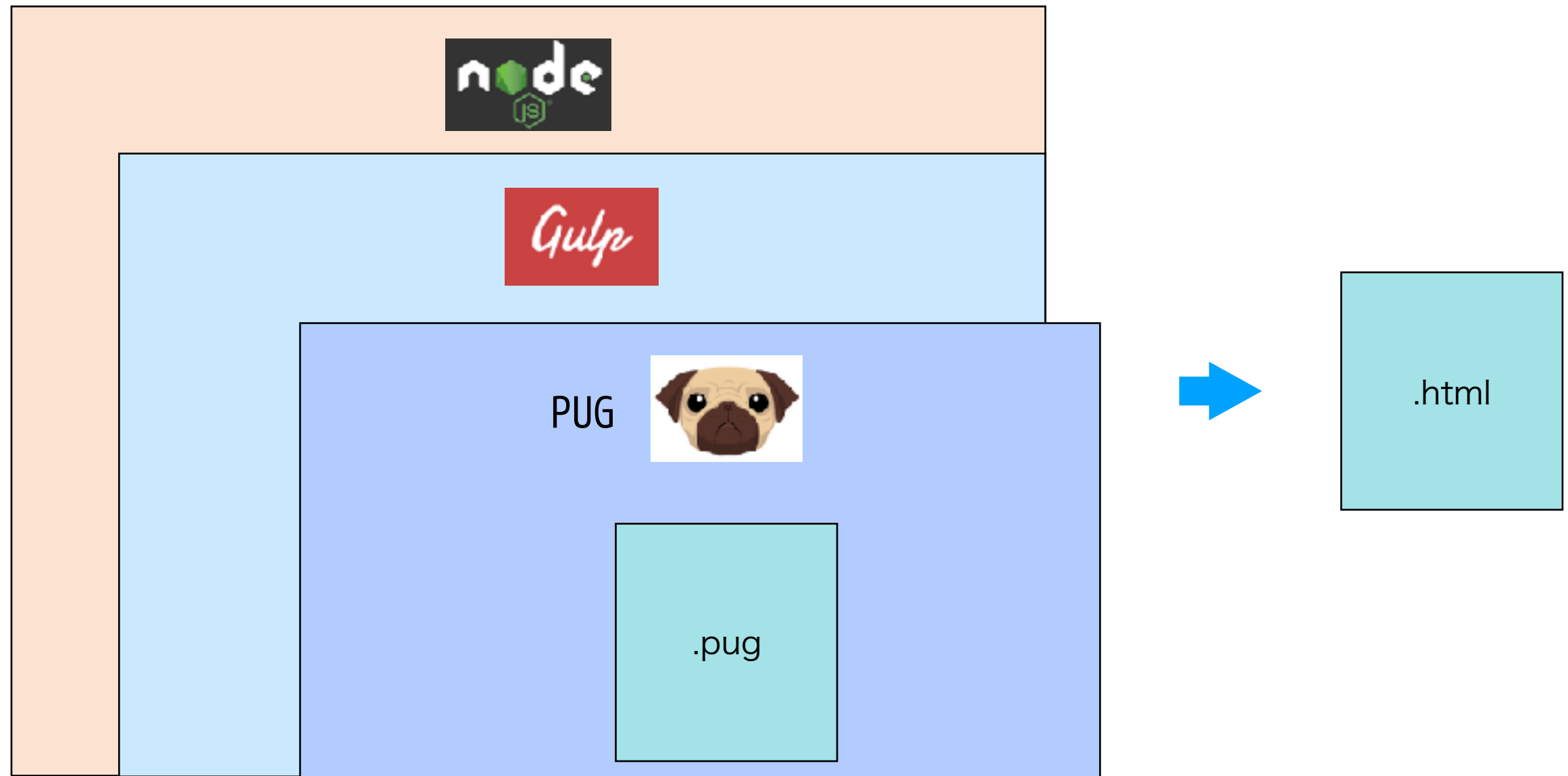
# Pugの特徴

---

- 以前はJadeという名称
- HTMLを簡単に書くためのHTMLを拡張したメタ言語
- 変数設定・継承・インクルードなどができる。
- pugファイル内でJavaScriptがつかえる。

# Pugを使うための準備

---



# Pugを使うための準備

---

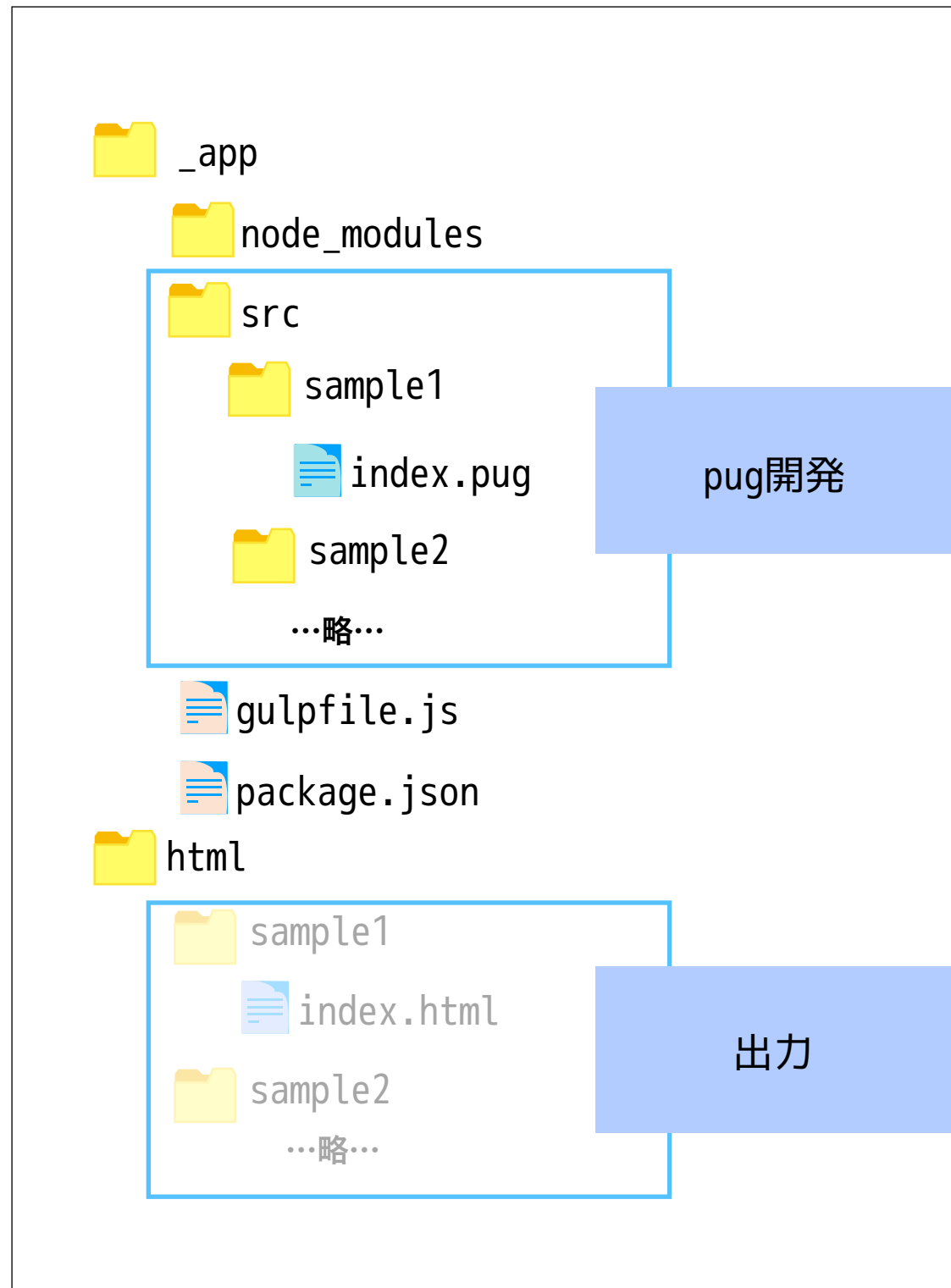
サンプルを用意しました。下記がリポジトリになります。

[https://github.com/asaandyoru/html5exam\\_docs](https://github.com/asaandyoru/html5exam_docs)

こちらから `git clone` またはダウンロードしてください。

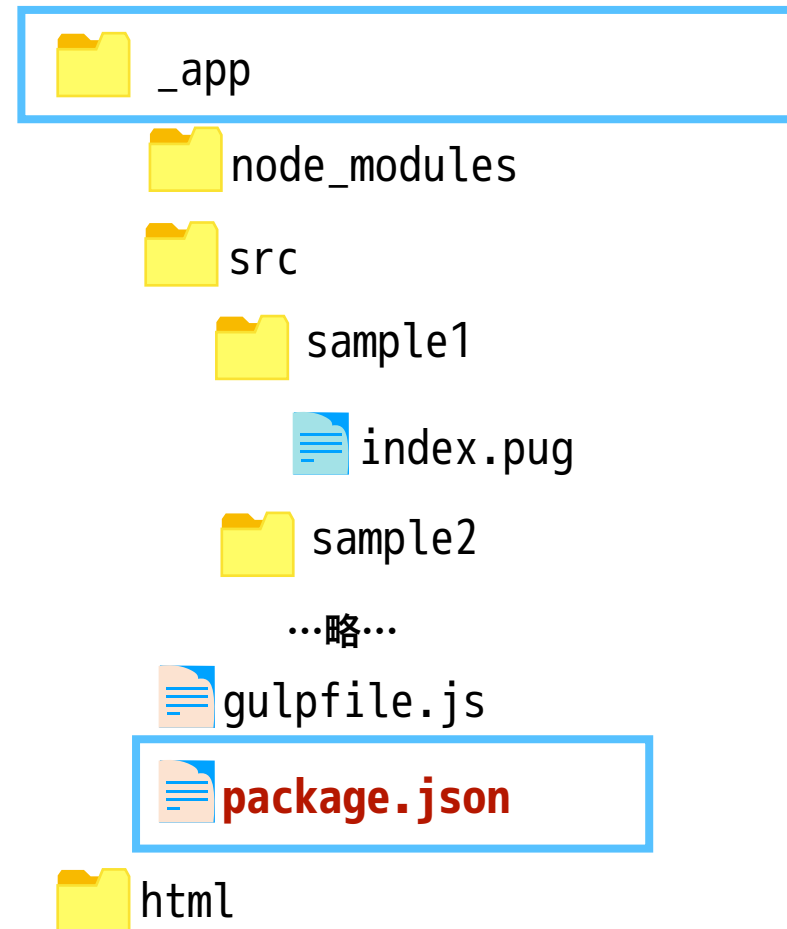
※node.jsのインストールは割愛させていただきます。

# サンプルPug構成



- 今回のサンプルの構成です
- `/_app/src/`以下がpugの開発エリア
- `/html/` 下 がドキュメントルート

# Pugを使うための準備

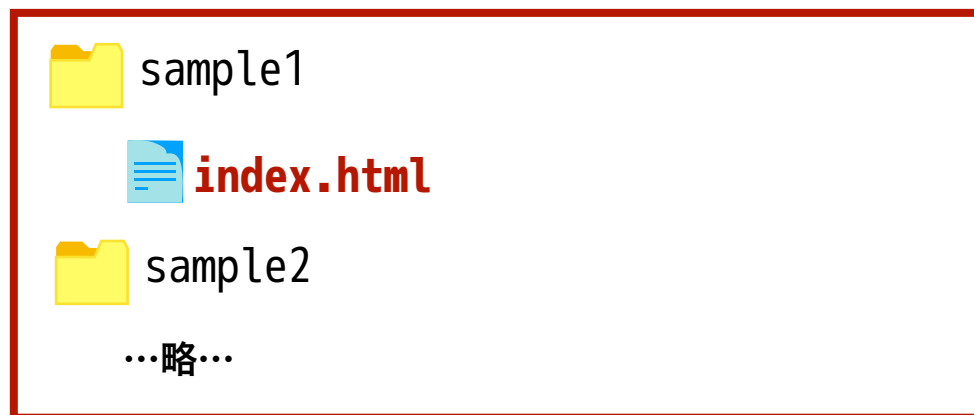
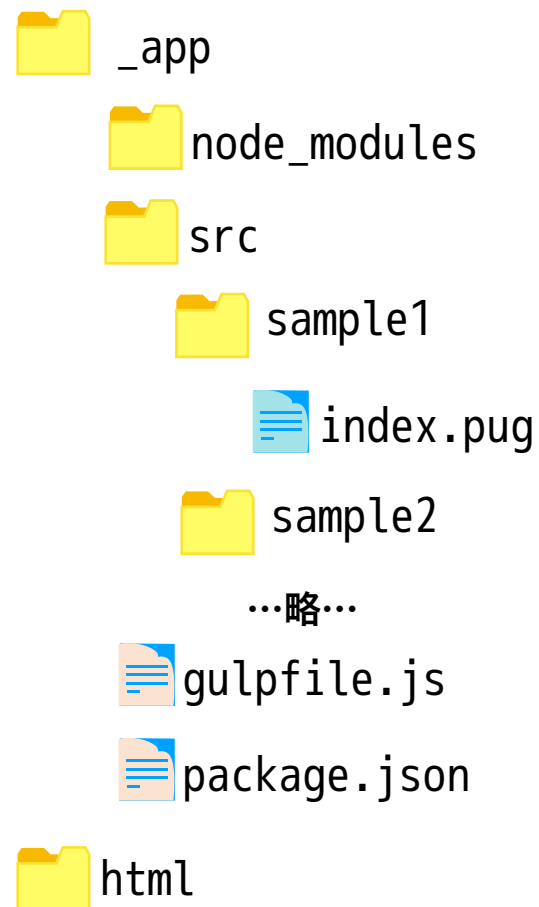


`package.json`がある/`_app`ディレクトリからコマンド入力しインストールできます。

```
$ cd html5exam_docs/_app
$ npm install
```



# Pugを使うための準備



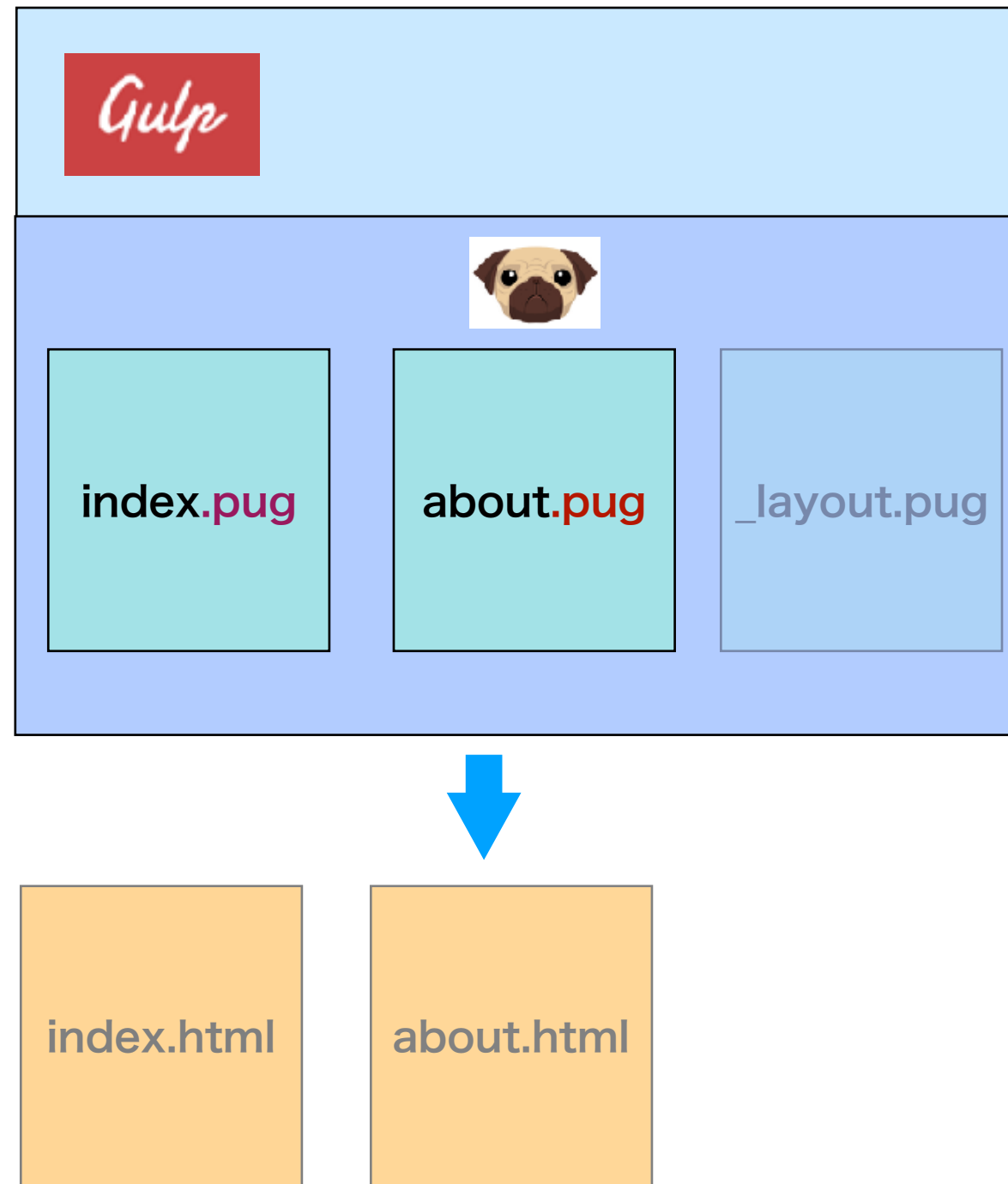
下記のコマンドを打つことでhtml  
ファイルが書き出されました。

```
$ npm run build
```

or

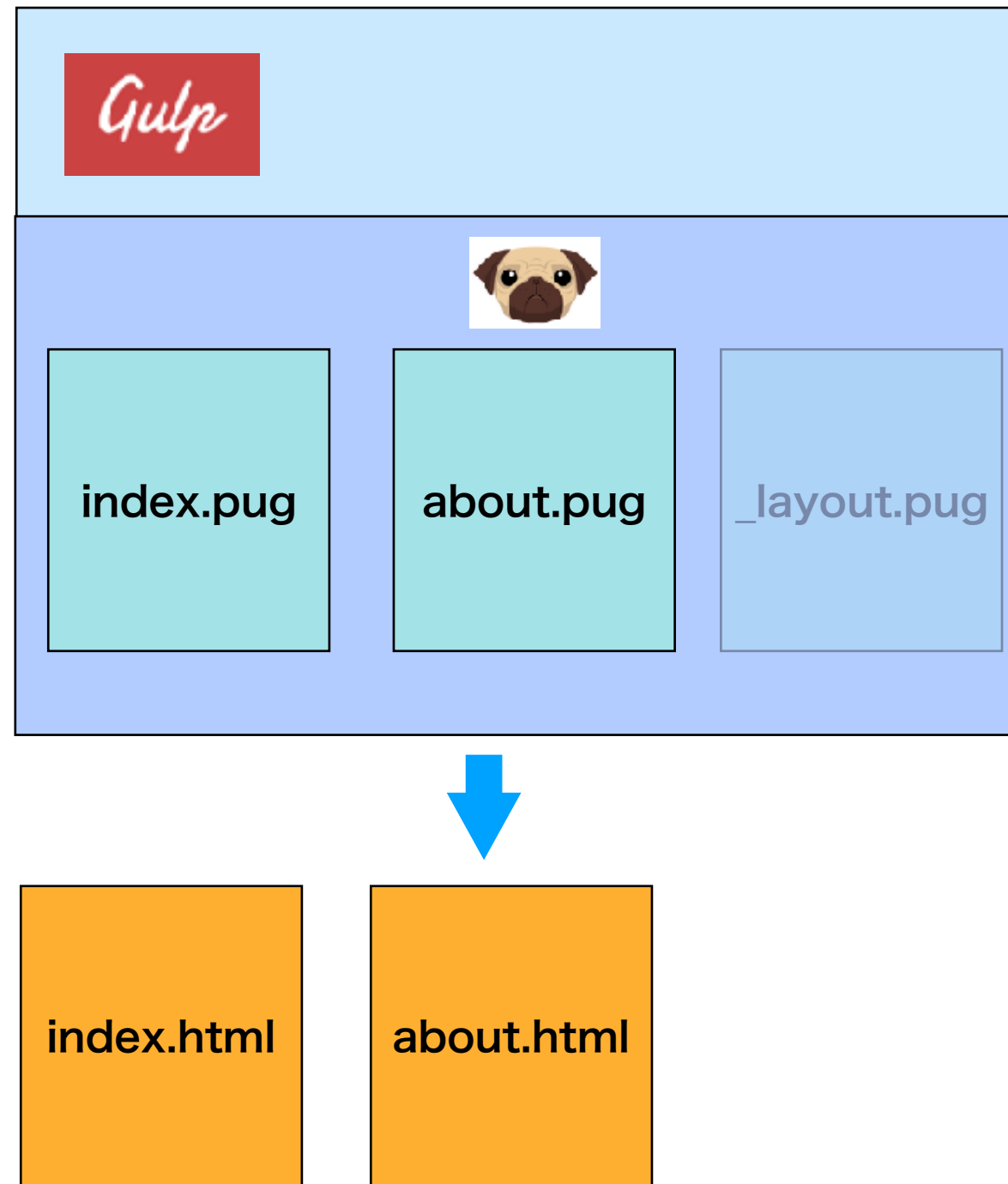
```
$ gulp build
```

# Pugを使うための準備



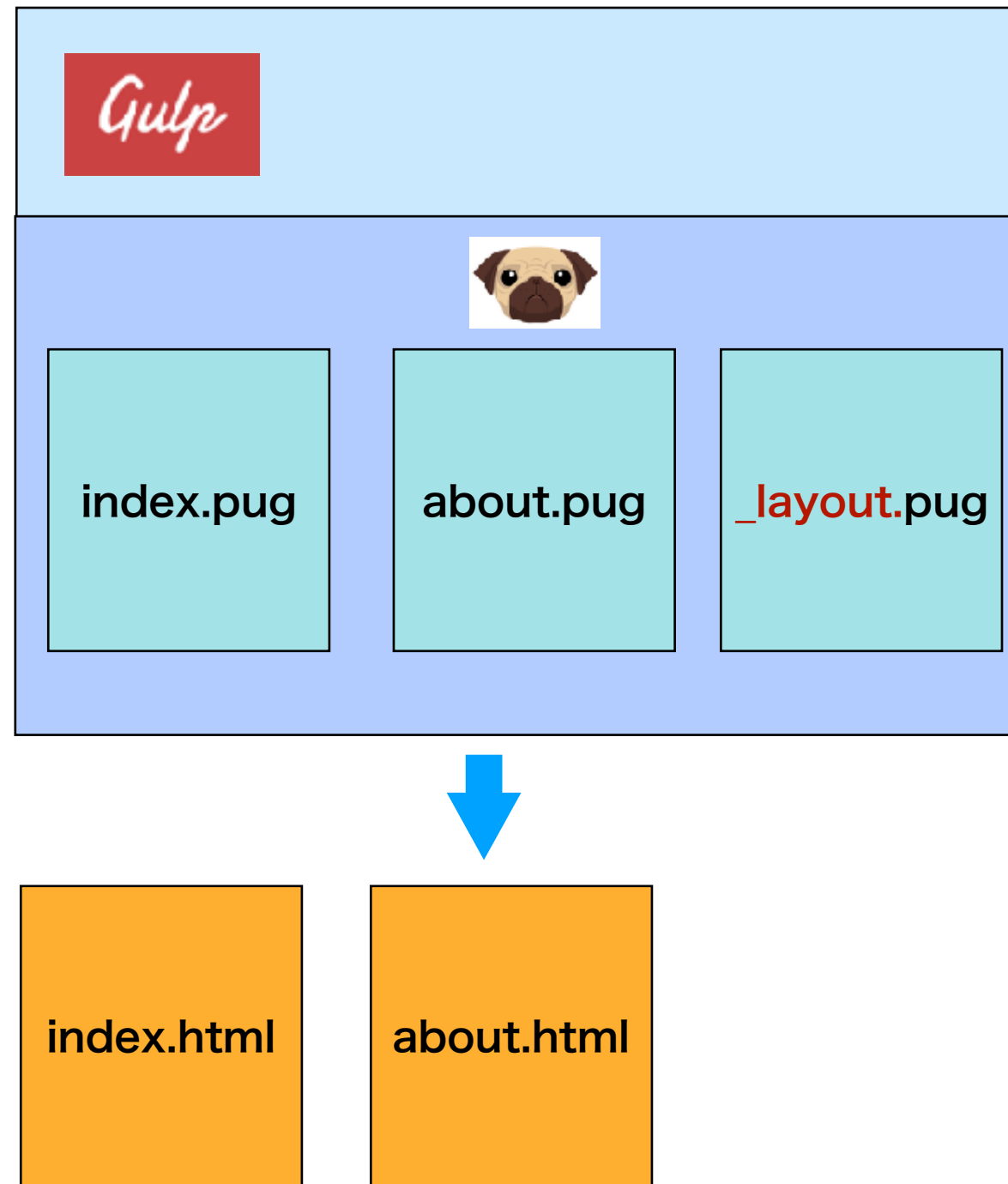
- 拡張子.pug

# Pugを使うための準備



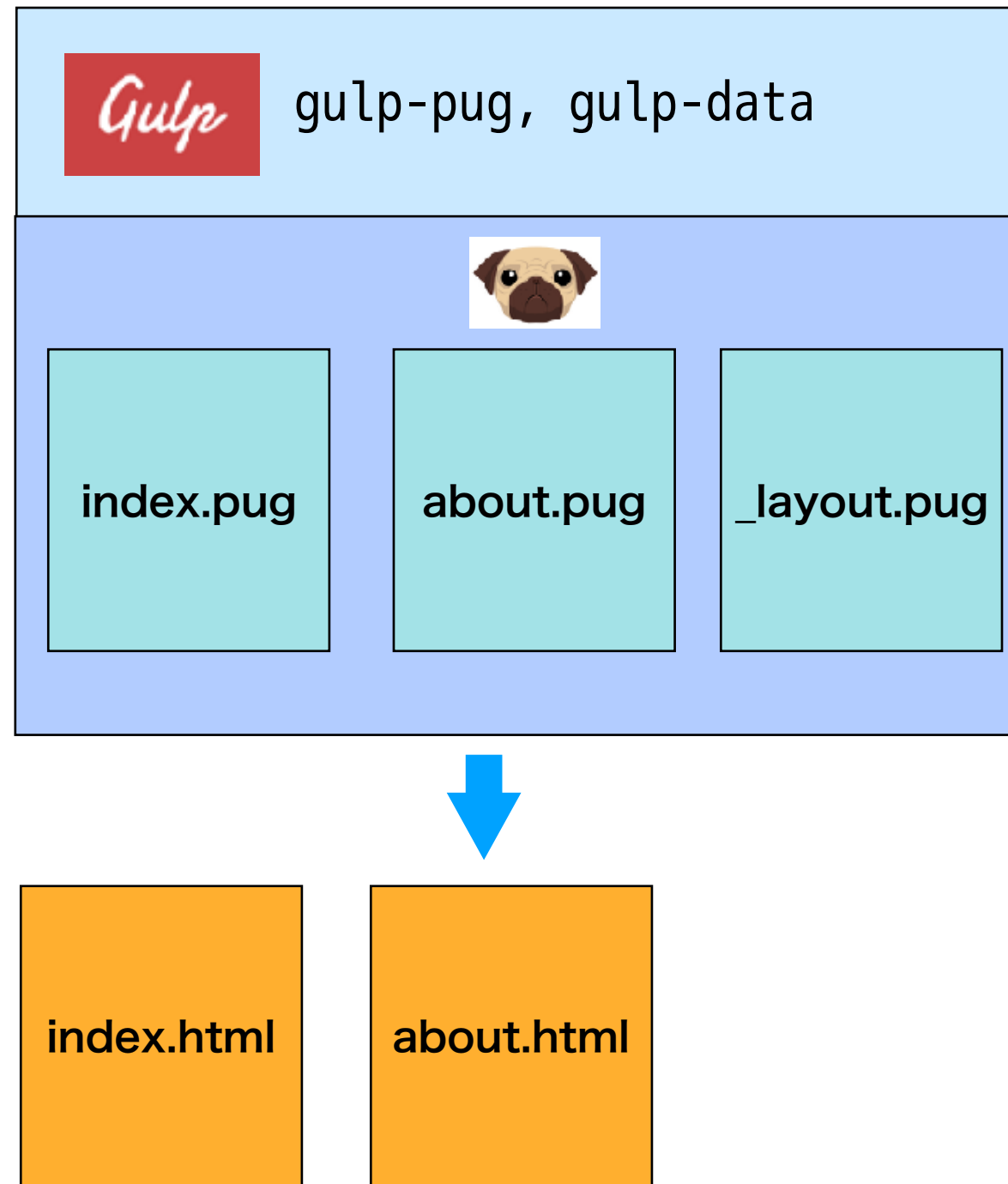
- 拡張子.pug
- 同じ名前の.htmlを出力

# Pugを使うための準備



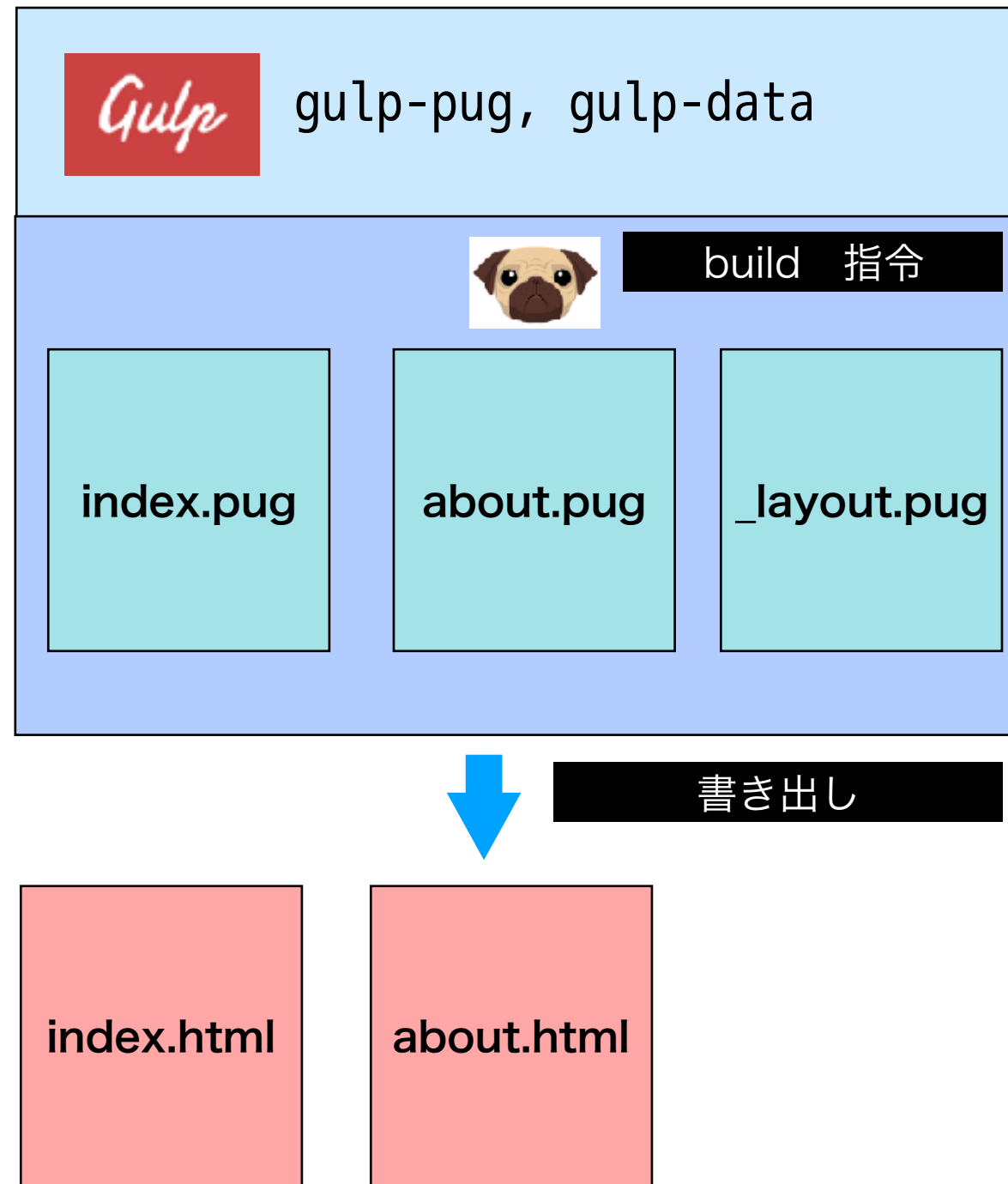
- 拡張子.pug
- 同じ名前の.htmlを出力
- 「\_」アンダースコア付きは書き出さない様にするのが普通

# Pugを使うための準備



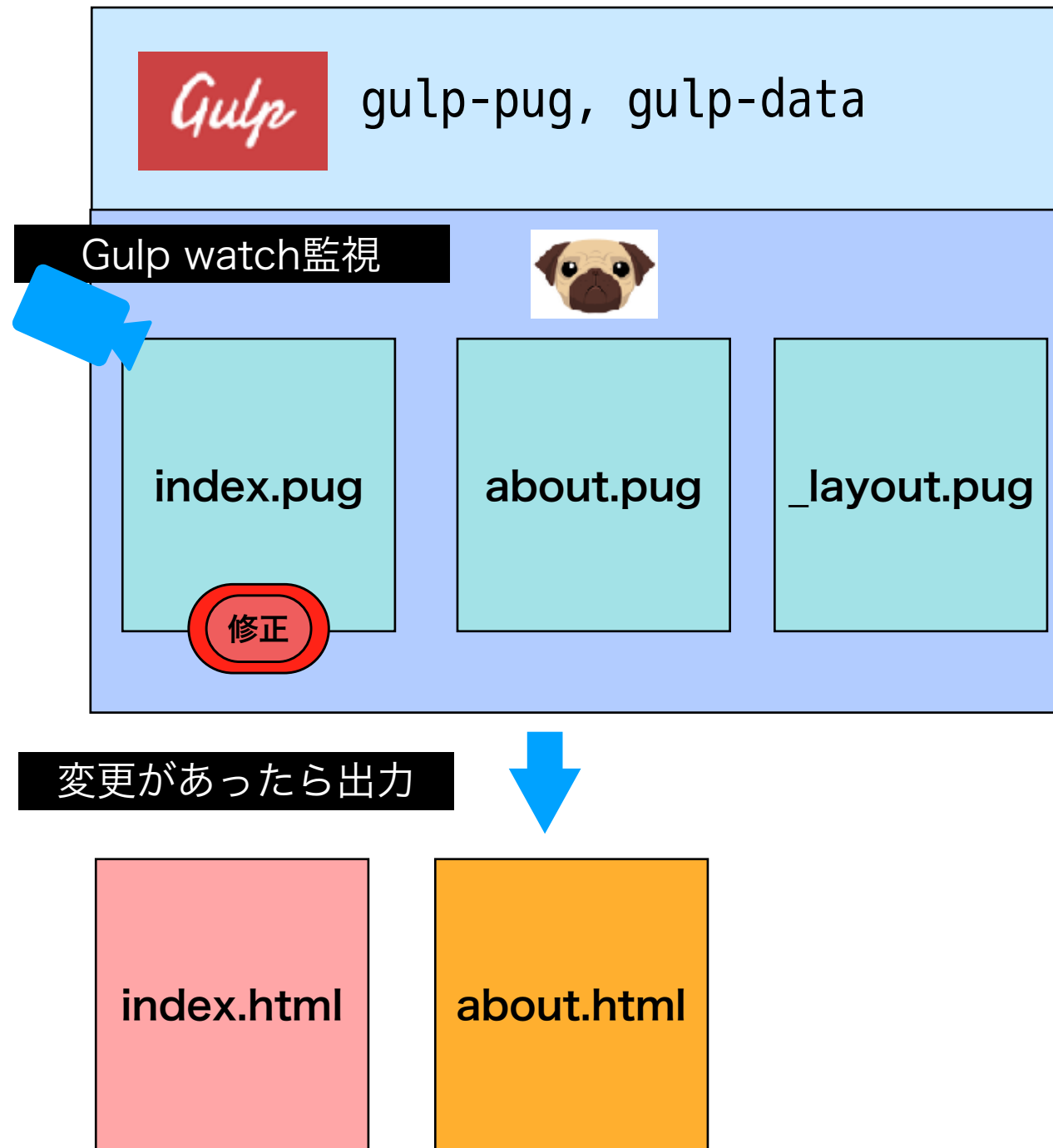
- 拡張子.pug
- 同じ名前の.htmlを出力
- 「\_」アンダースコア付きは書き出さない様にするのが普通
- gulp-pug, gulp-data moduleを使う。

# Pugを使うための準備



- 拡張子.pug
- 同じ名前の.htmlを出力
- 「\_」パーシャル付きは書き出さない  
ようにするのが普通
- gulp-pug, gulp-data moduleを使う
- 全てを書き出すコマンド  
npm run build または gulp build

# Pugを使うための準備



- 拡張子.pug
- 同じ名前の.htmlを出力
- 「\_」パーシャル付きは書き出さないようにするのが普通
- gulp-pug, gulp-data moduleを使う
- 全てを書き出すコマンド  
npm run build または gulp build
- npm run watch または gulp watch コマンドで.pugの変更を監視し.htmlに出力

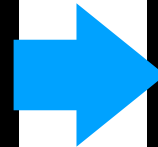
# Pugの基本



# Pugの特徴 記法

index.pug 📁 /\_app/src/sample1

```
doctype
html(lang="ja")
  head
    title Pug
  body
    h1 Pug Examples
    div.container
      p
        a(href="/about.html")
          | Pug example
```




index.html 📁 /html/sample1

```
<!DOCTYPE html>
<html lang="ja">
  <head>
    <title>Pug</title>
  </head>
  <body>
    <h1>Pug Examples</h1>
    <div class="container">
      <p>
        <a href="/about.html">
          Pug example
        </a>
      </p>
    </div>
  </body>
</html>
```

# Pugの特徴 変数

index.pug

 /\_app/src/sample2


head

```
- var mytitle= "Pugの特徴";
```

```
title= mytitle
```



index.html

 /html/sample2

```
<head>
```

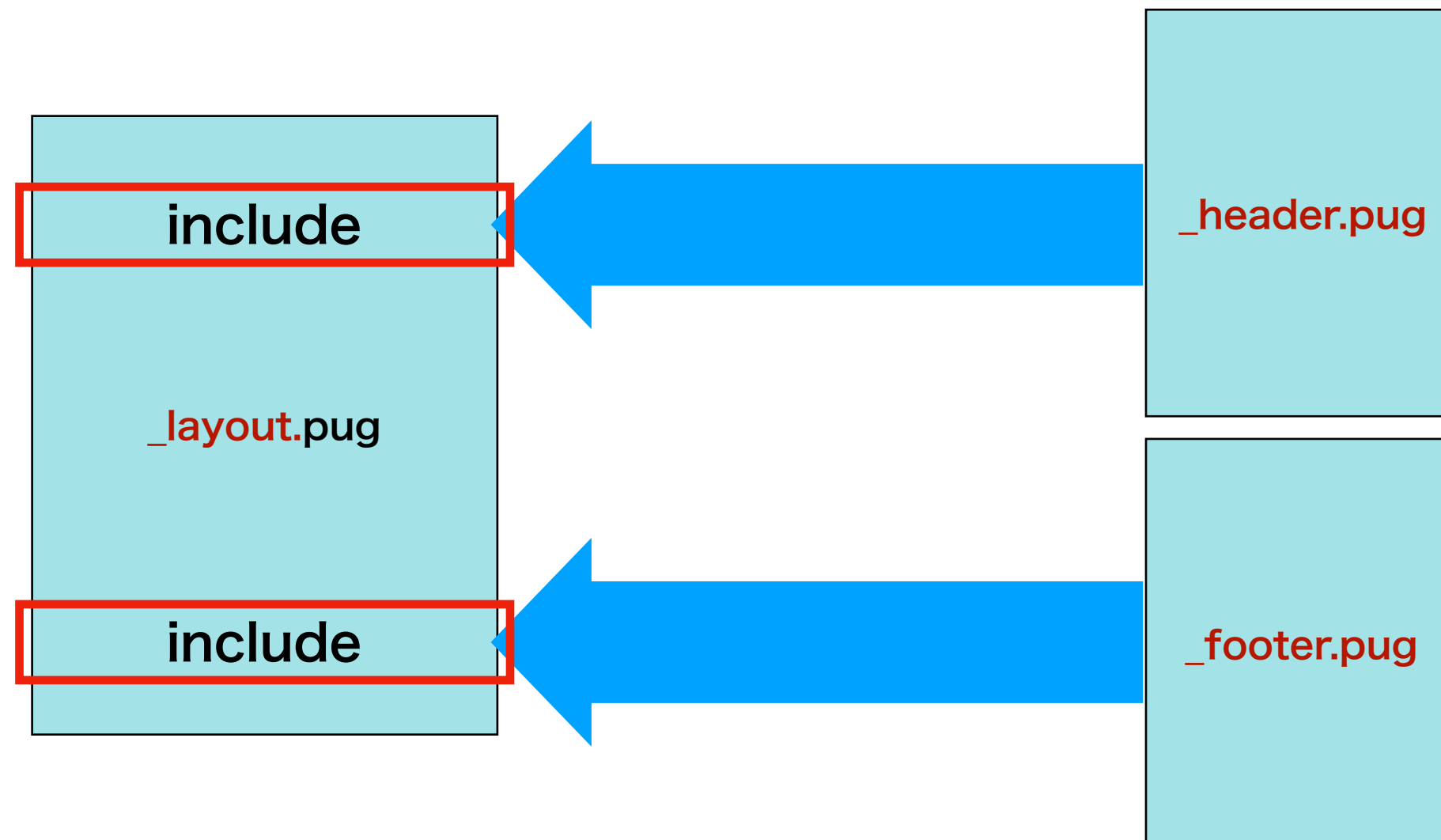
```
  <title>Pugの特徴</title>
```

```
</head>
```

# Pugの特徴 インクルードとは

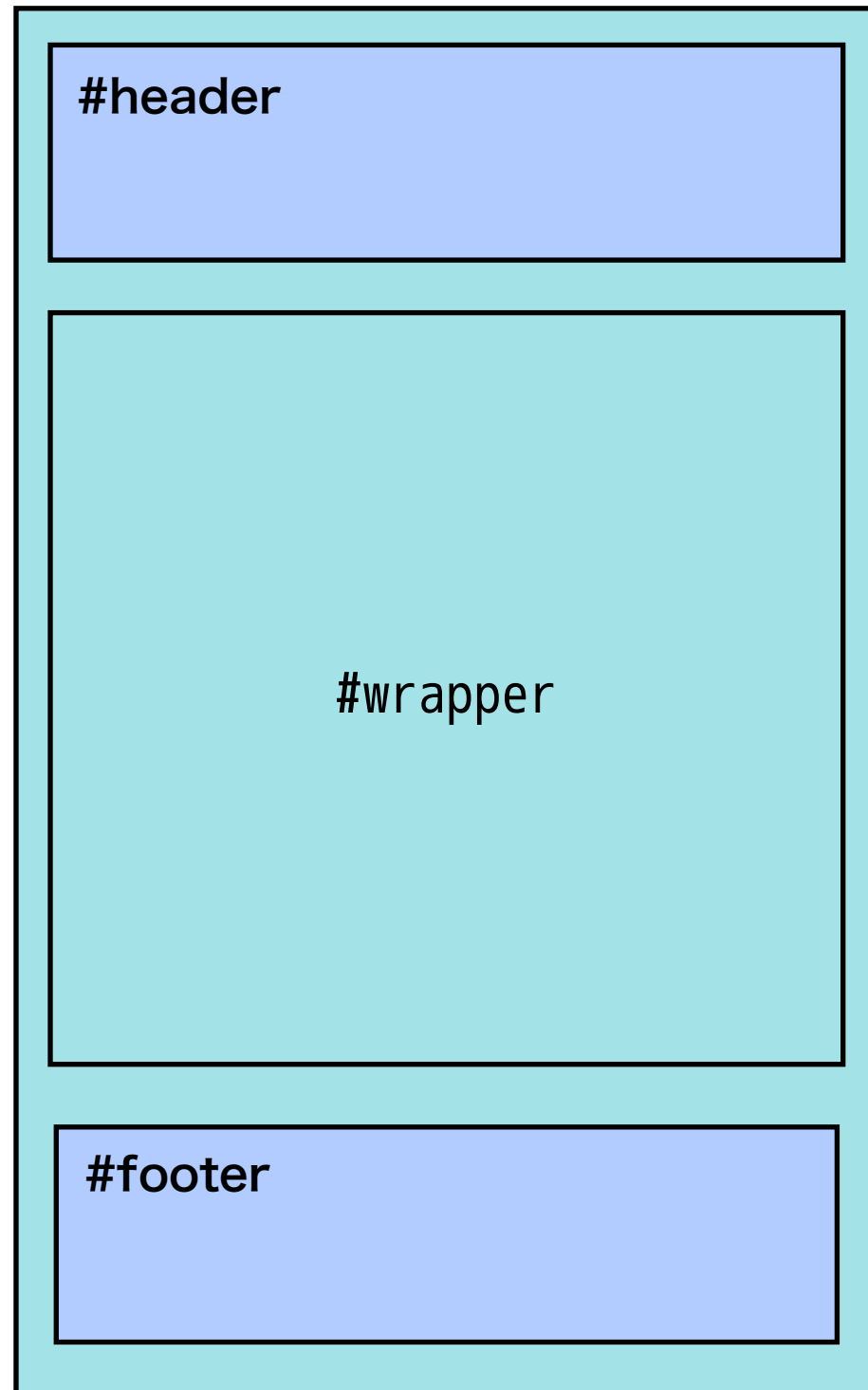
---

インクルードとは別ファイルに記述したソースコードを読み込む機能です。




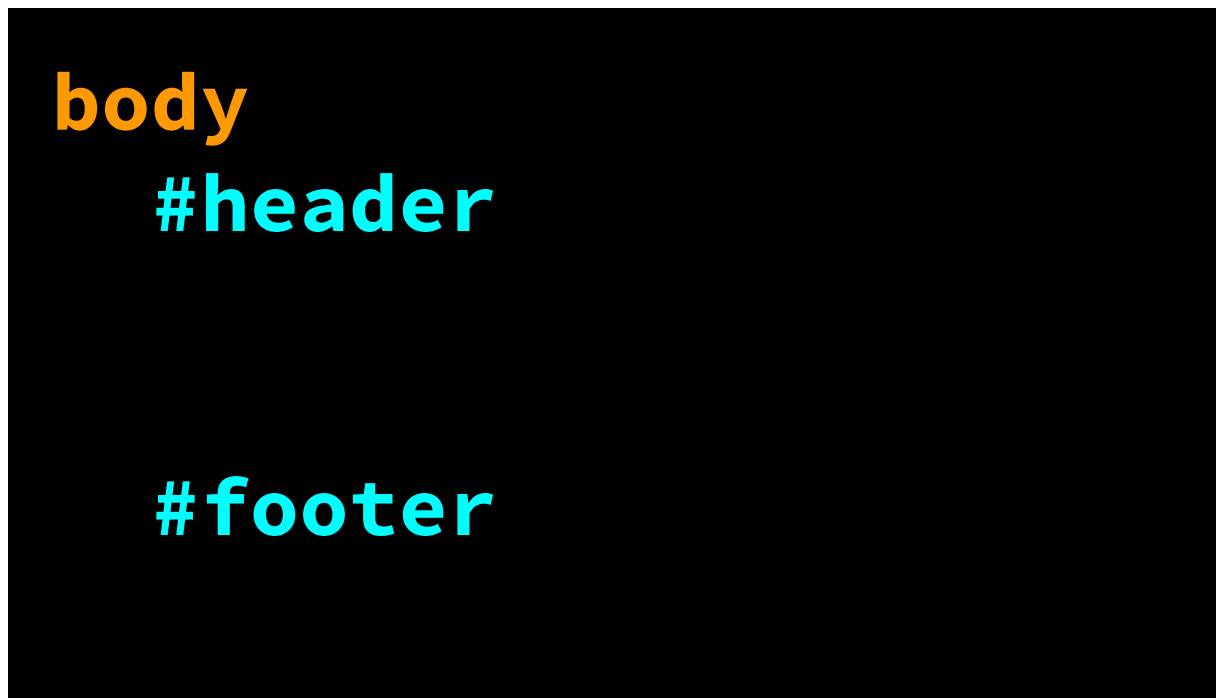
# Pugの特徴 インクルード

index.pug



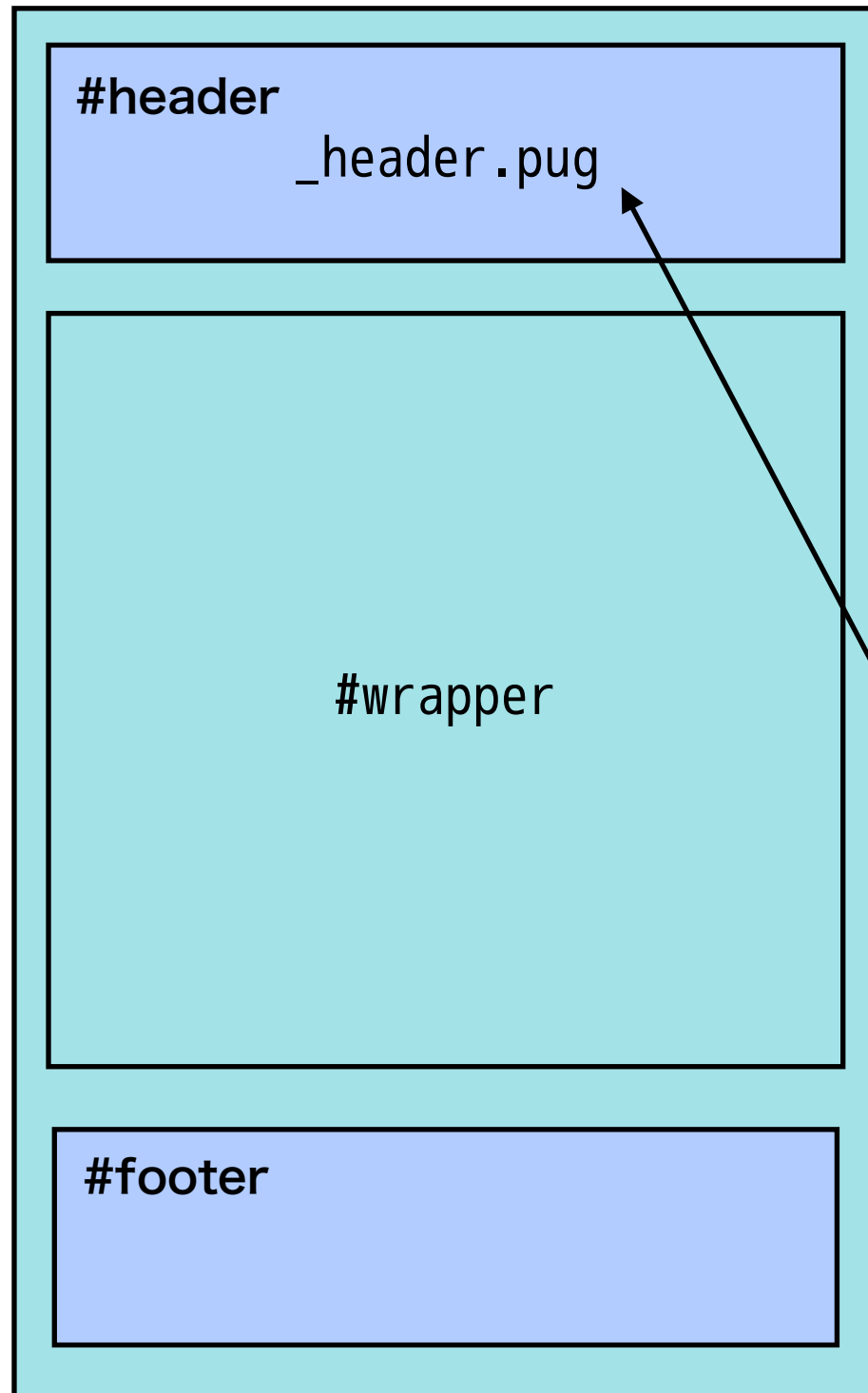
index.pug

 /\_app/src/sample3




# Pugの特徴 インクルード

index.pug



index.pug

 /\_app/src/sample3

```
body
  #header
    include _header.pug

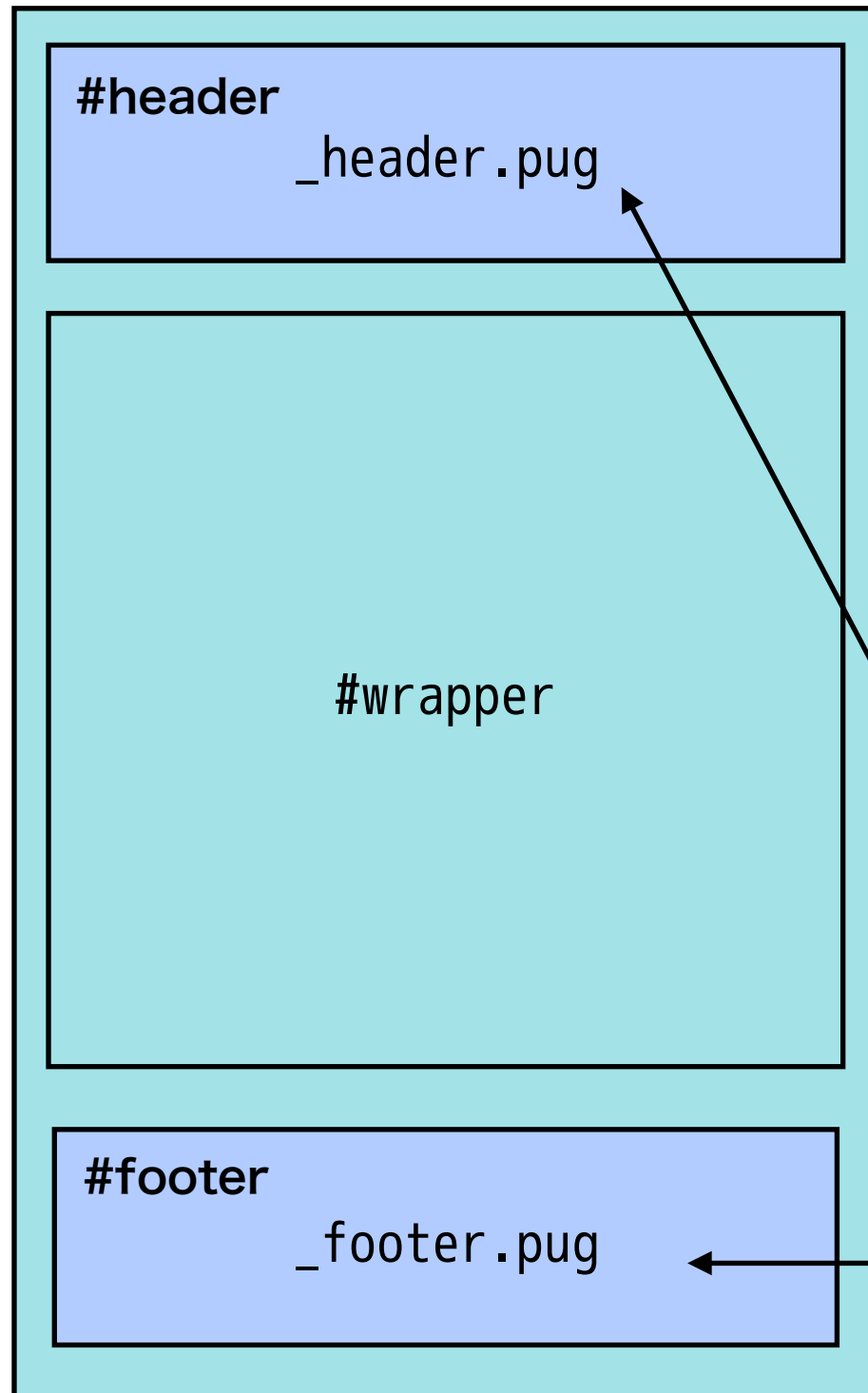
  #footer
```

\_header.pug


```
p Pugの特徴
```

# Pugの特徴 インクルード

index.pug



index.pug

 /\_app/src/sample3

```
body
  #header
    include _header.pug

  #footer
    include _footer.pug
```

\_header.pug

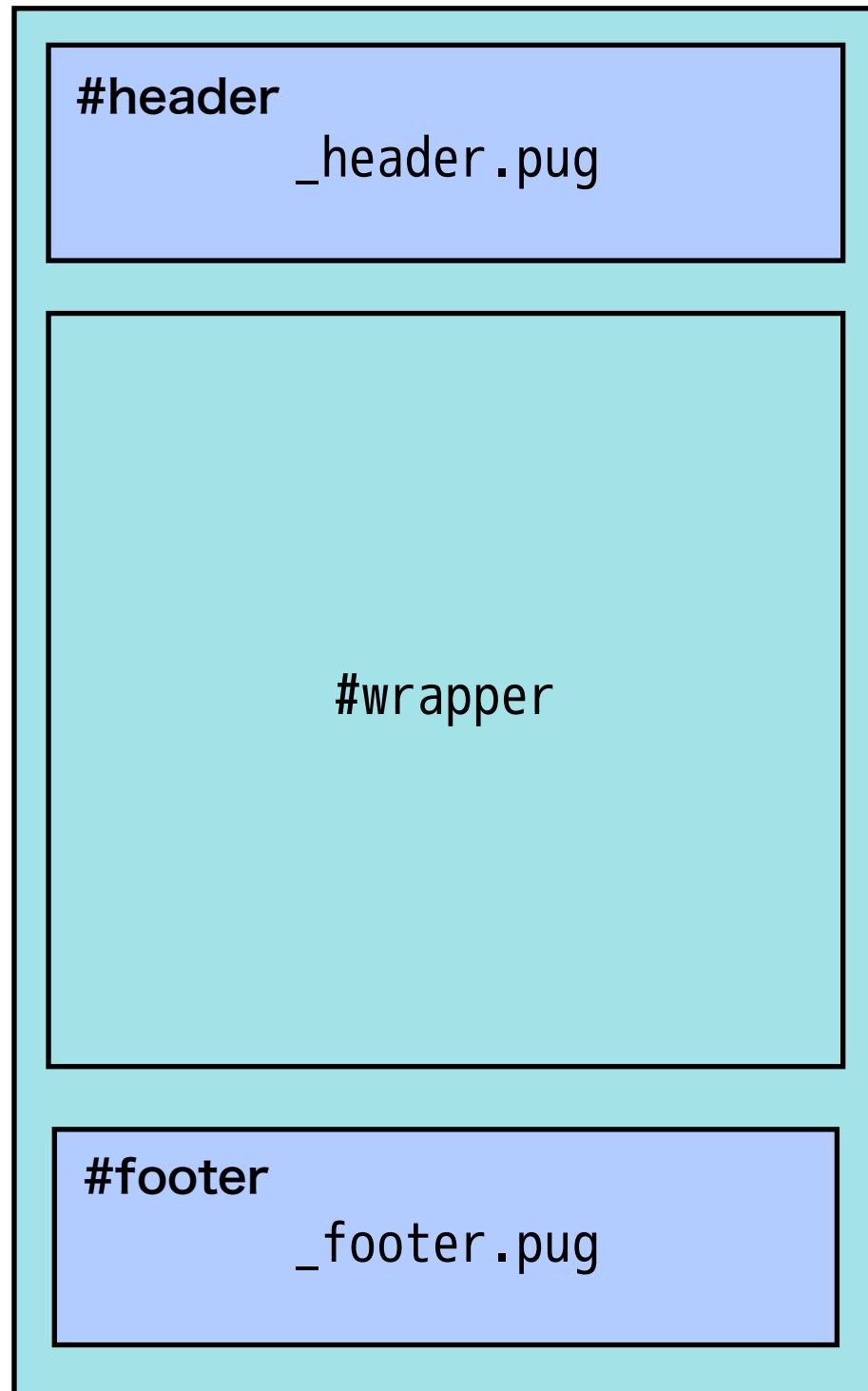
```
p Pugの特徴
```

\_footer.pug


```
p copyright
```

# Pugの特徴 インクルード

index.pug



index.html

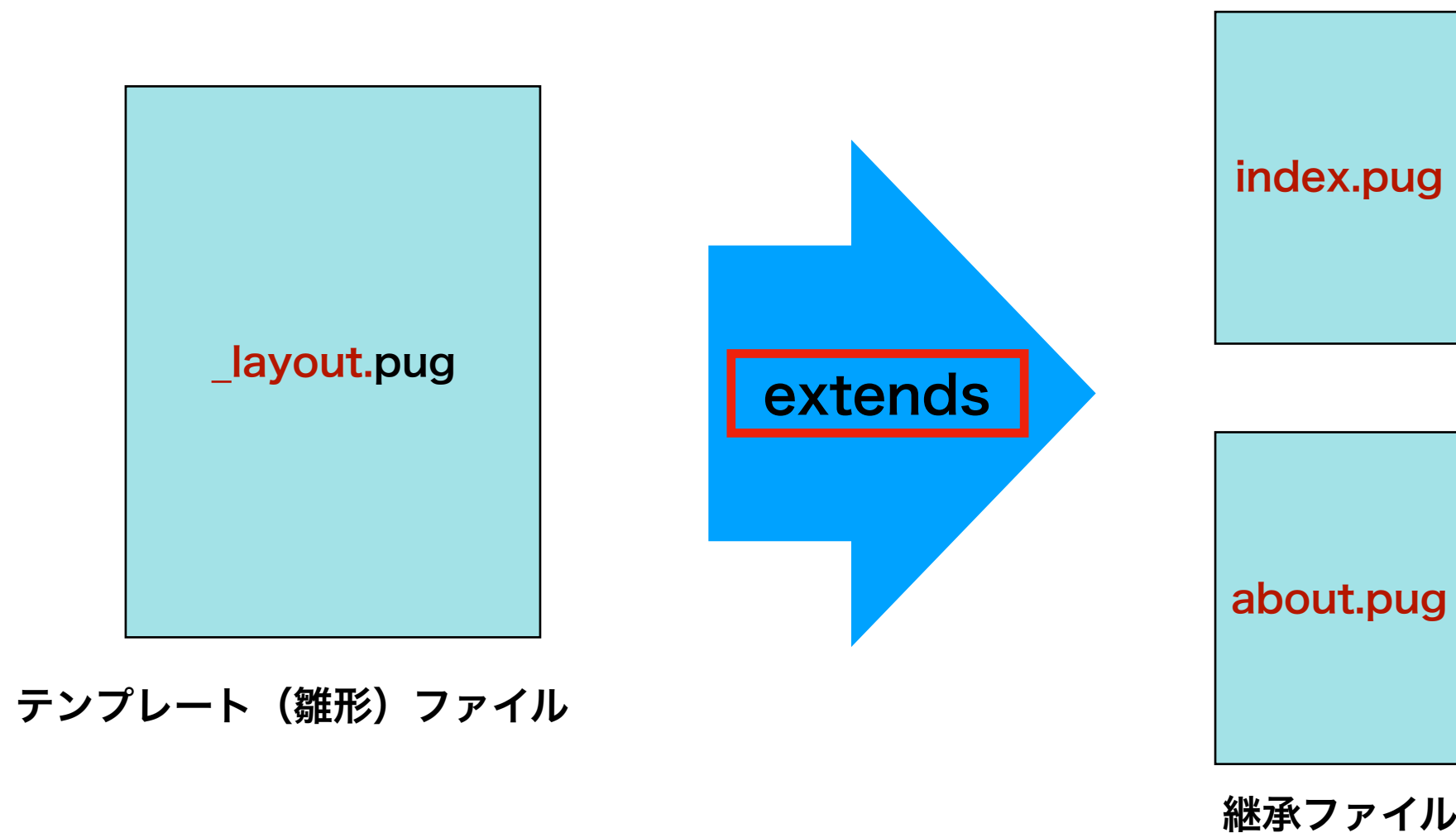
 /\_app/src/sample3

```
<body>
  <div id="header">
    <p>Pugの特徴</p>
  </div>
  <div id="footer">
    <p>copyright</p>
  </div>
</body>
```

# Pugの特徴 継承とは

---


継承とはテンプレートのコードを活かしつつ、新規ファイルにあらたにコードを上書きしたり、書きかえたり、追加したりする機能

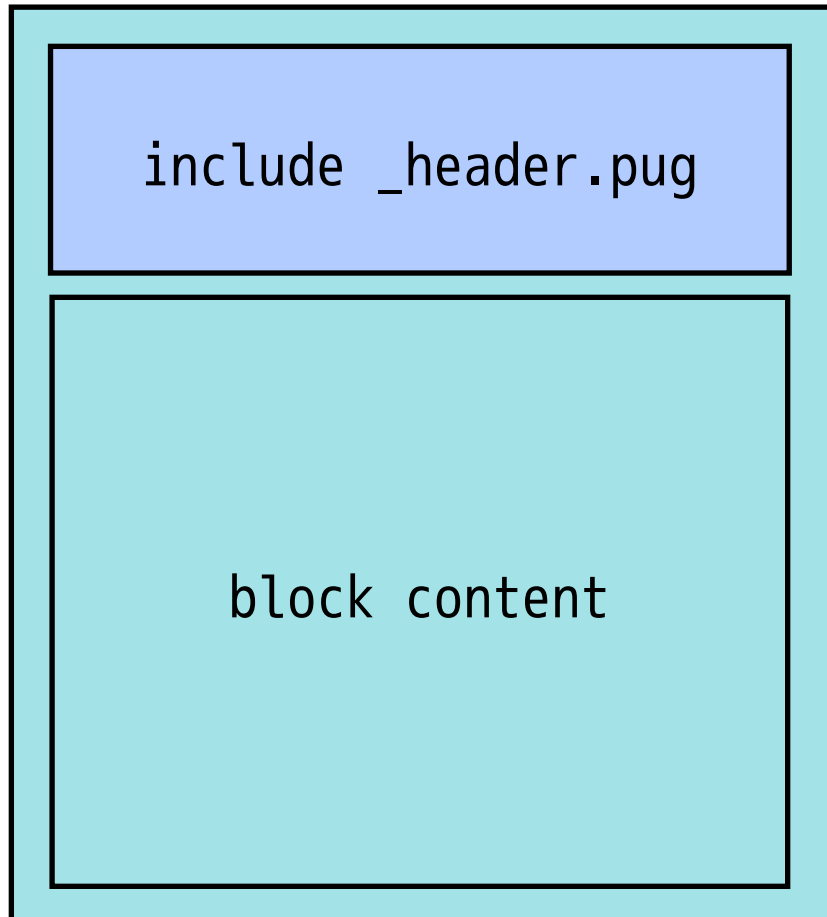




# Pugの特徴 継承 テンプレートファイル作成

`_layout.pug`

 `/_app/src/sample4`



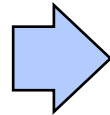
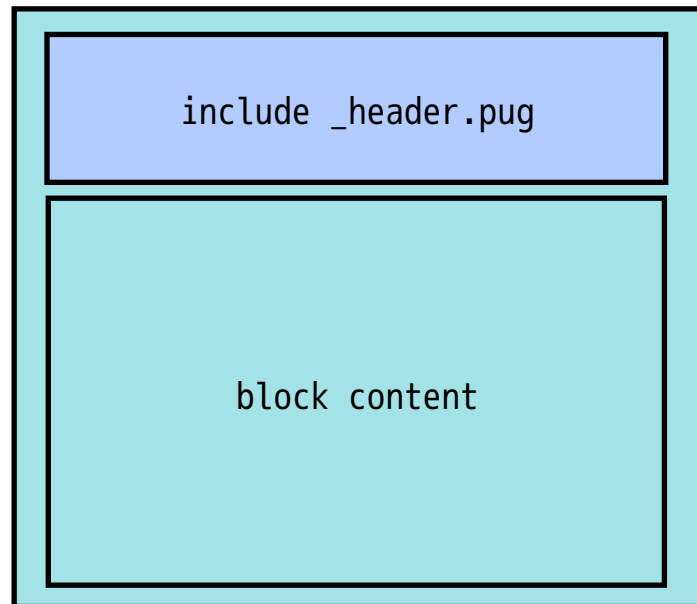
```
head
  title Pugの特徴

body
  include _header.pug

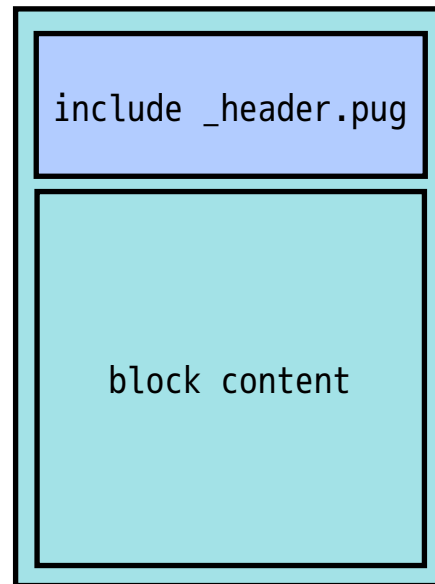
  block content
    p I have a pen
```


# Pugの特徴 継承

**\_layout.pug**  /\_app/src/sample4



**index.pug**

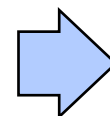


 /\_app/src/sample4

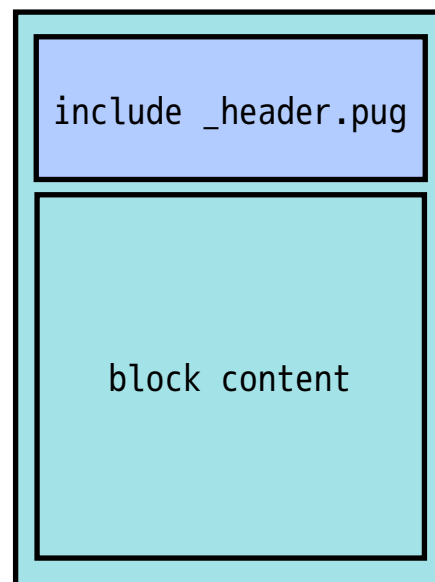
```
extends _layout
block content
  p I have an apple.
```


```
head
  title Pugの特徴
body
  include _header

block content
  p I have a pen
```



**about.pug**

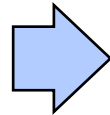
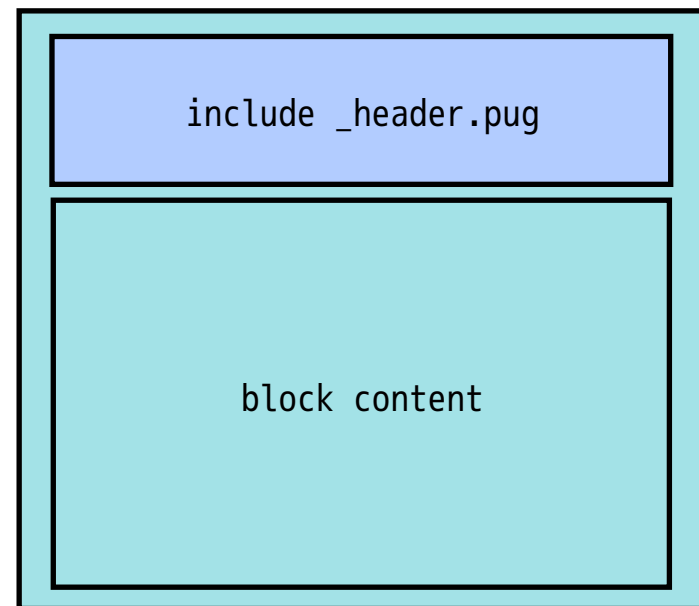


 /\_app/src/sample4

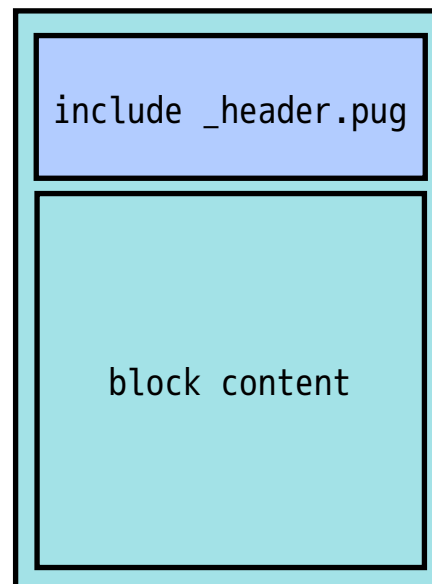
```
extends _layout
block content
  p I have a pineapple.
```


# Pugの特徴 継承

**\_layout.pug**  /\_app/src/sample4



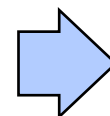
**index.pug**



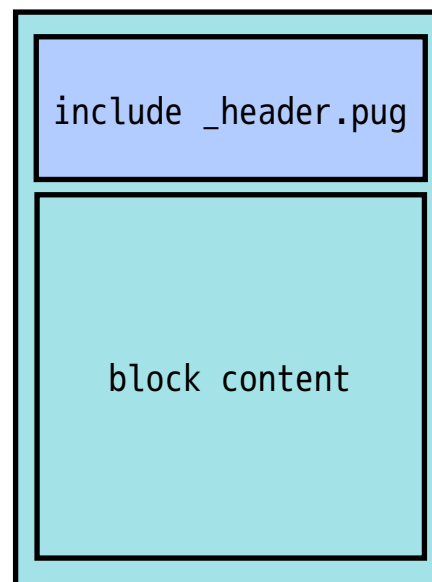
 /\_app/src/sample4


```
extends _layout
block content
  p I have an apple.
```

```
head
  title Pugの特徴
body
  include _header
  block content
    p I have a pen
```



**about.pug**

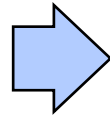
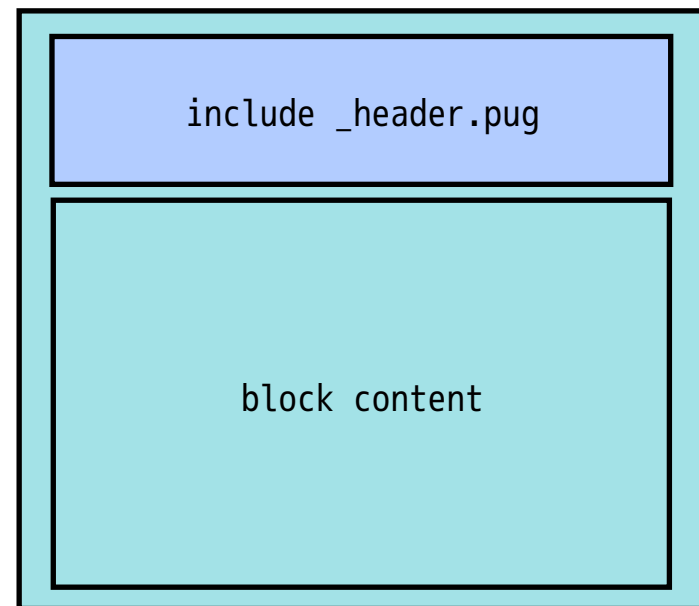


 /\_app/src/sample4

```
extends _layout
block content
  p I have a pineapple.
```

# Pugの特徴 継承

**\_layout.pug**  /\_app/src/sample4



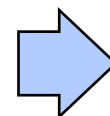
**index.html**

 /html/sample4

```
<head>
  <title>Pugの特徴</title>
</head>
<body>
  <p>ピコ太郎</p>
  <p>I have an apple.</p>
</body>
```

```
head
  title Pugの特徴
body
  include _header

  block content
    p I have a pen
```



**about.html**

 /html/sample4

```
<head>
  <title>Pugの特徴</title>
</head>
<body>
  <p>ピコ太郎</p>
  <p>I have a pineapple.</p>
</body>
```

# Pugの特徴 まとめ

---

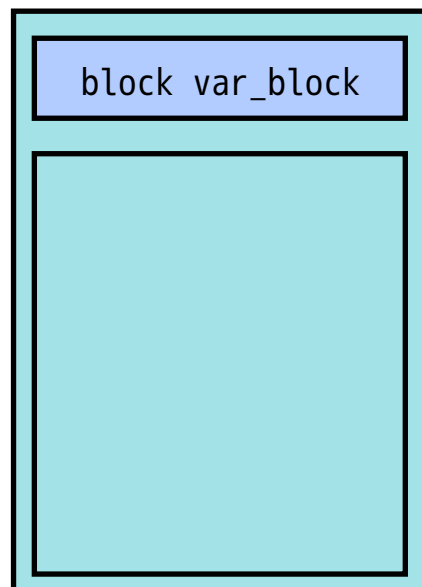
- タグを省略できる記述
- 変数が見える
- インクルードが見える
- 「extends」と「block」をつかうことで継承できる

今回は時間の関係上省略しますが mixinやloop処理などこの他にも強力なものもあります興味があれば調べてみてください。


# Pug応用編

# Pug応用編 データをページ内にまとめる

\_layout.pug



## デフォルトの変数設定

 /\_app/src/sample5

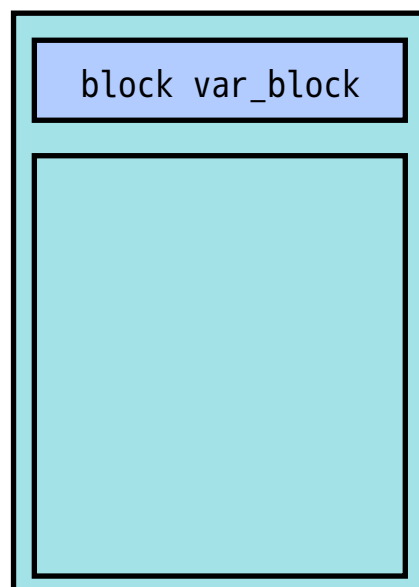
```
//-. default.global.var
- var title = "デフォルトタイトル";
- var description = "デフォルトディスクリプション";
- var keywords = "デフォルトキーワード";

//-. var.override
block var_block


doctype
html(lang="ja")
  head
    meta(charset="utf-8")
    meta(name="description", content=description)
    meta(name="keywords", content=keywords)
    title!=title
```

# Pug応用編 データをページ内にまとめる

\_layout.pug



## デフォルトの変数設定

 /\_app/src/sample5

```
//- default.global.var
- var title = "デフォルトタイトル";
- var description = "デフォルトディスクリプション";
- var keywords = "デフォルトキーワード";

//- var.override
block var_block

doctype
html(lang="ja")
  head
    meta(charset="utf-8")
    meta(name="description", content=description)
    meta(name="keywords", content=keywords)
    title!=title
```

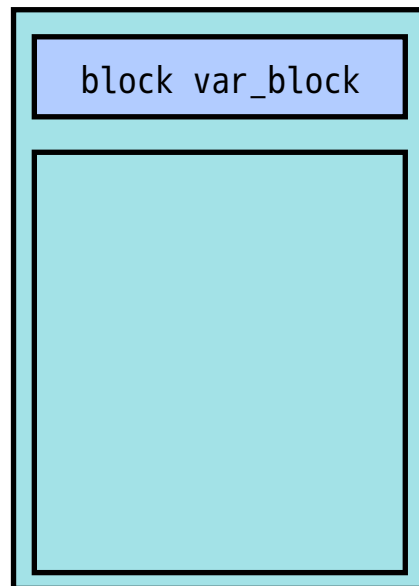
変数用block



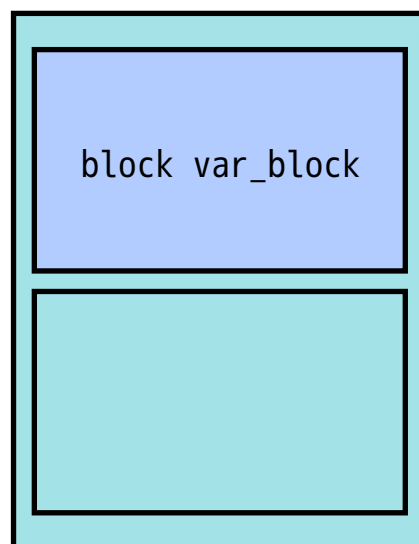
# Pug応用編 データをページ内にまとめる

/\_app/src/sample5

\_layout.pug



index.pug



```
//- default.global.var
- var title = "デフォルトタイトル";
- var description = "デフォルトディスクリプション";
- var keywords = "デフォルトキーワード";

//- var.override
block var_block

doctype
html(lang="ja")
head
  meta(charset="utf-8")
  meta(name="description", content=description)
  meta(name="keywords", content=keywords)
  title!=title
```

**extends \_layout**

**block var\_block**

```
- title = "TOP タイトル";
- description = "TOP ディスクリプション";
- keywords = "TOP キーワード";
```

**blockを使って変数を上書き**

# Pug応用編 データをページ内にまとめる

ページ上部にまとめられているので管理しやすい

index  
.pug

```
extends _layout

block var_block
  - title = "TOP タイトル";
  - description = "TOP ディスクリプション";
  - keywords = "TOP キーワード";
```

xxx  
.pug

```
extends _layout

block var_block
  - title = "TOP タイトル";
  - description = "TOP ディスクリプション";
  - keywords = "TOP キーワード";
```

xxx  
.pug

```
extends _layout

block var_block
  - title = "TOP タイトル";
  - description = "TOP ディスクリプション";
  - keywords = "TOP キーワード";
```

xxx  
.pug

```
extends _layout

block var_block
  - title = "TOP タイトル";
  - description = "TOP ディスクリプション";
  - keywords = "TOP キーワード";
```

# Pug応用編 データをページ内にまとめる

ただしページそれぞれにデータを入れる必要がある  
-> データを**外部ファイル**にまとめたい。

index  
.pug

```
extends _layout

block var_block
  - title = "TOP タイトル";
  - description = "TOP ディスクリプション";
  - keywords = "TOP キーワード";
```

xxx  
.pug

```
extends _layout

block var_block
  - title = "TOP タイトル";
  - description = "TOP ディスクリプション";
  - keywords = "TOP キーワード";
```

xxx  
.pug

```
extends _layout

block var_block
  - title = "TOP タイトル";
  - description = "TOP ディスクリプション";
  - keywords = "TOP キーワード";
```

xxx  
.pug

```
extends _layout

block var_block
  - title = "TOP タイトル";
  - description = "TOP ディスクリプション";
  - keywords = "TOP キーワード";
```

# Pug応用編 外部jsonファイルについて

```
{  
  "default" : {  
    "og_url" : "http://www.fork.co.jp",  
    "og_title" : "株式会社フォーク",  
    "og_image" : "http://www.fork.co.jp/img/og.png"},  
  "local": {  
    "/sample6/index.html" : {  
      "title" : "TOP | 株式会社フォーク",  
      "description" : "TOPページ"  
    },  
    "/sample6/about/index.html" : {  
      "title" : "会社概要 | 株式会社フォーク",  
      "description" : "会社概要"  
    }  
  }  
}
```

 /\_app/src/data.json

# Pug応用編 外部jsonのデータの持ち方を工夫する

```
{  
  "default" : {  
    "og_url" : "http://www.fork.co.jp",  
    "og_title" : "株式会社フォーク",  
    "og_image" : "http://www.fork.co.jp/img/og.png"},  
  "local": {  
    "/sample6/index.html" : {  
      "title" : "TOP | 株式会社フォーク",  
      "description" : "TOPページ"},  
    "/sample6/about/index.html" : {  
      "title" : "会社概要 | 株式会社フォーク",  
      "description" : "会社概要"}  
  }  
}
```

デフォルトを設定

出力先のファイル名をキー

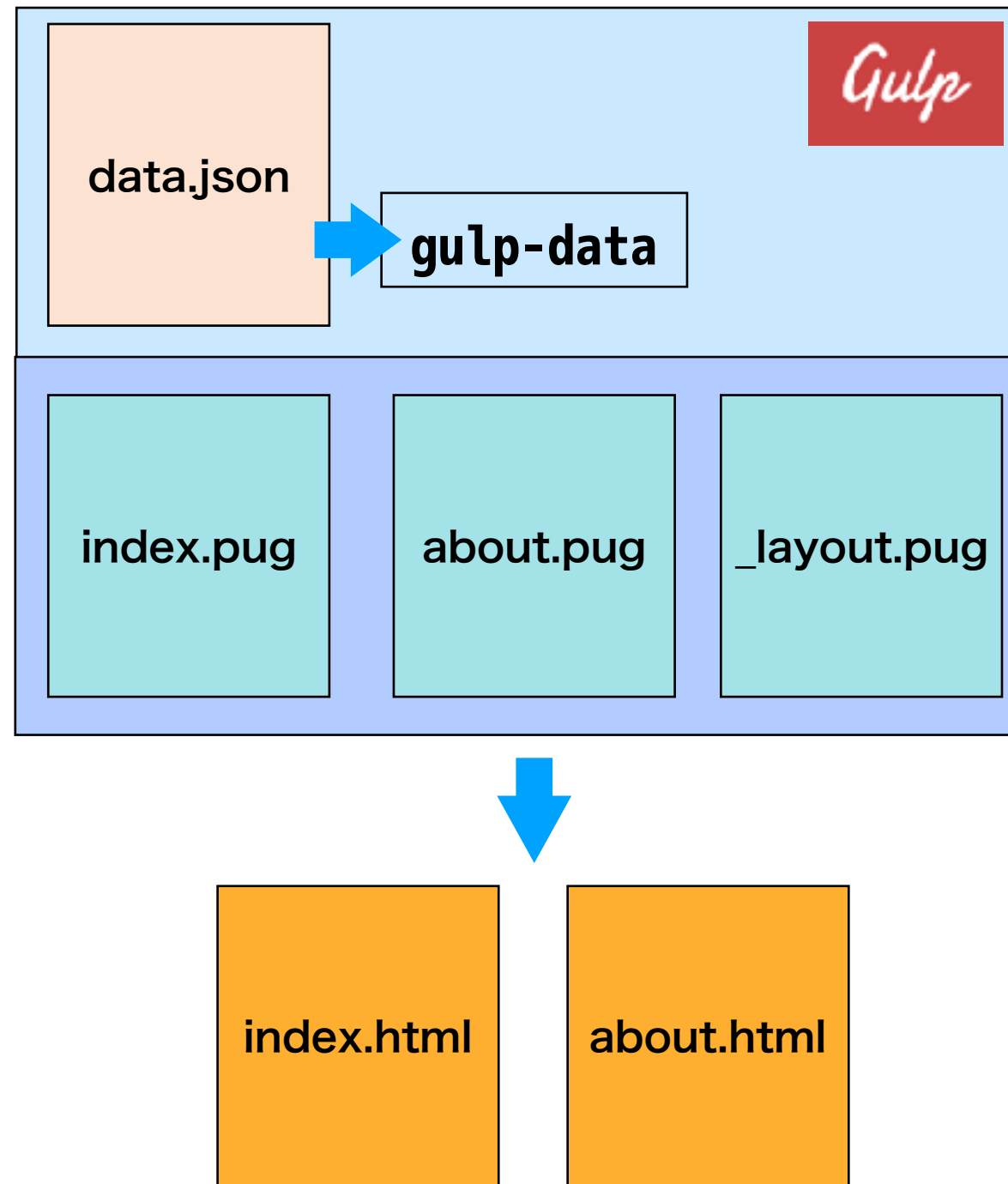
データ

ファイル名をキー

データ

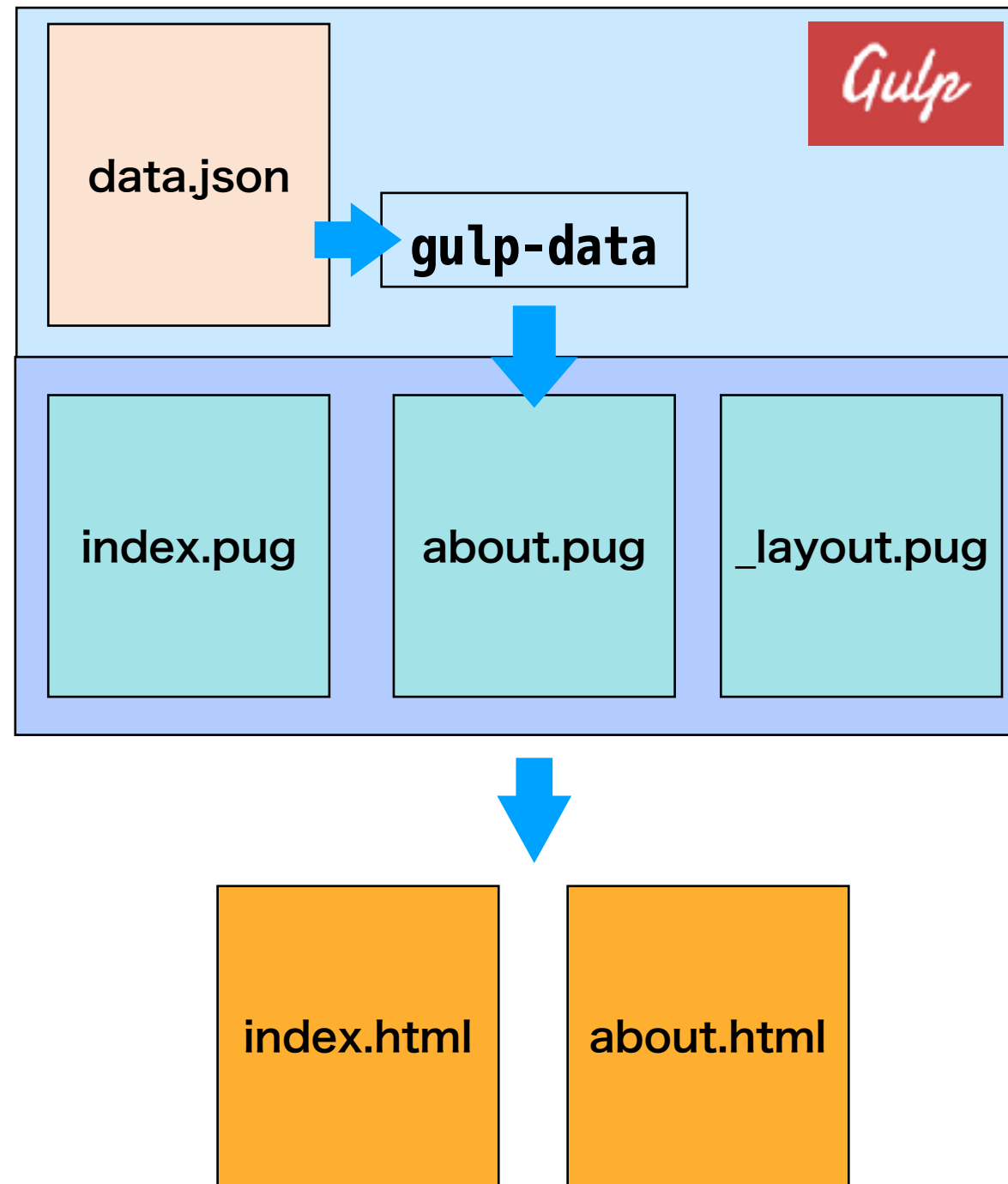
/\_app/src/data.json

# Pug応用編 外部jsonを利用する



- gulp-data moduleを使う

# Pug応用編 外部jsonを利用する



- gulp-data moduleを使う
- gulp-dataから.pugの**ファイルパス**と**外部jsonファイルのデータ**を渡しhtmlへ書き出す

# Pug応用編 外部jsonを利用する

```
- var title           = config.local.title;
- var description      = config.local.description;
- var keywords         = config.local.keywords;
- var og_url          = config.default.og_url;
- var og_title         = config.default.og_title;
- var og_image         = config.default.og_image;
```

**block** var\_block

doctype

**html**(lang="ja")

**head**

```
title=title
meta(name="description", content=description)
meta(name="keywords", content=keywords)
meta(property='og:url', content=og_url)
meta(property='og:title', content=og_title)
meta(property='og:image', content=og_image)
```

**body**



# Pug応用編 オンラインデータ管理

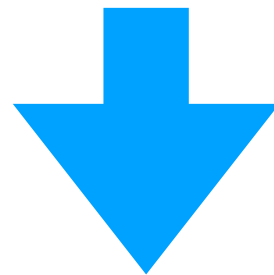
---

オンラインでデータ管理するメリット

# Pug応用編 オンラインデータ管理

---

オンラインでデータ管理するメリット



オンラインで情報を一元管理することで、ディレクター・デザイナーなどスタッフの誰でも編集できそのままhtml作成に利用できる。

# Pug応用編 データのスプレッドシート管理

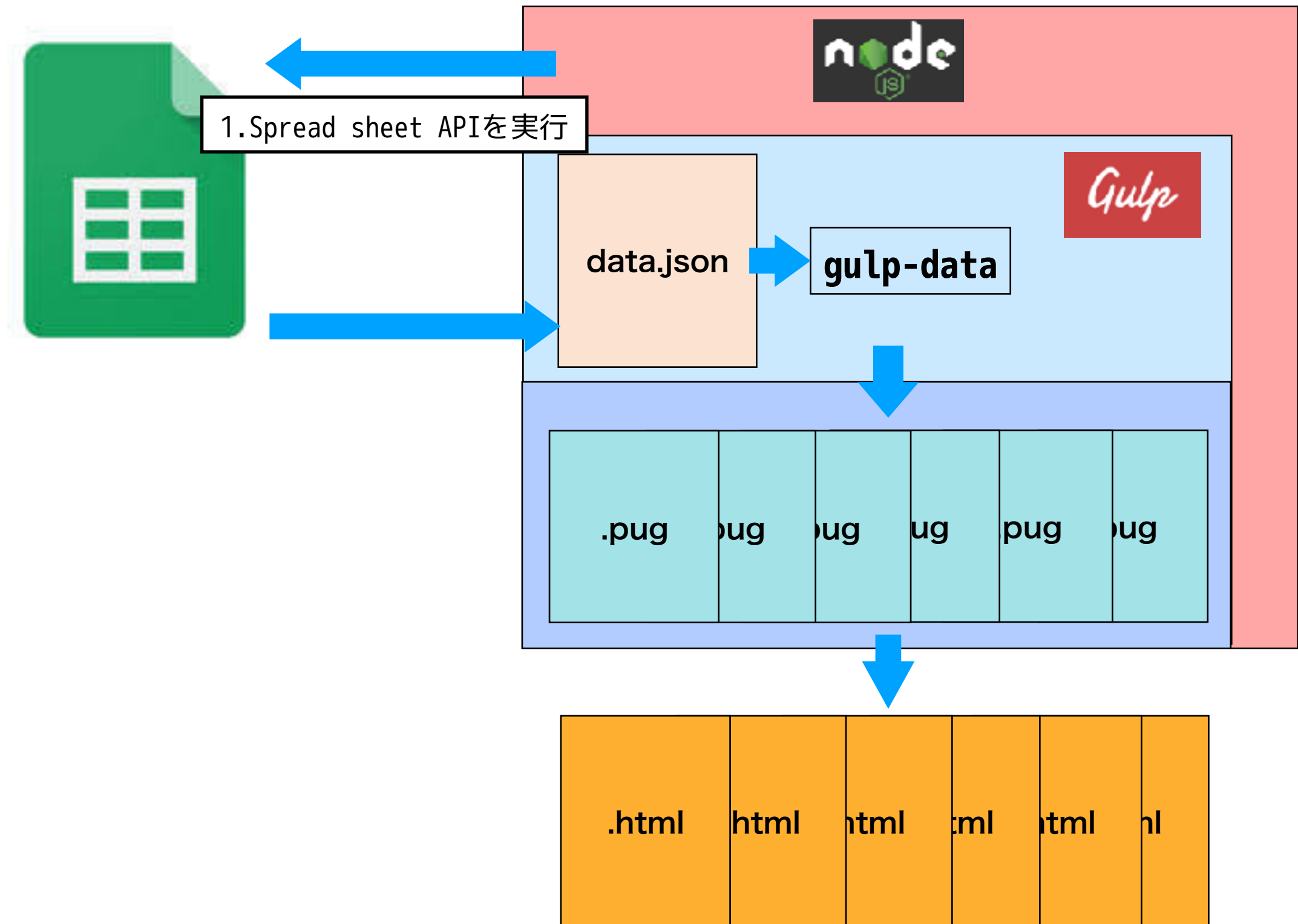
---



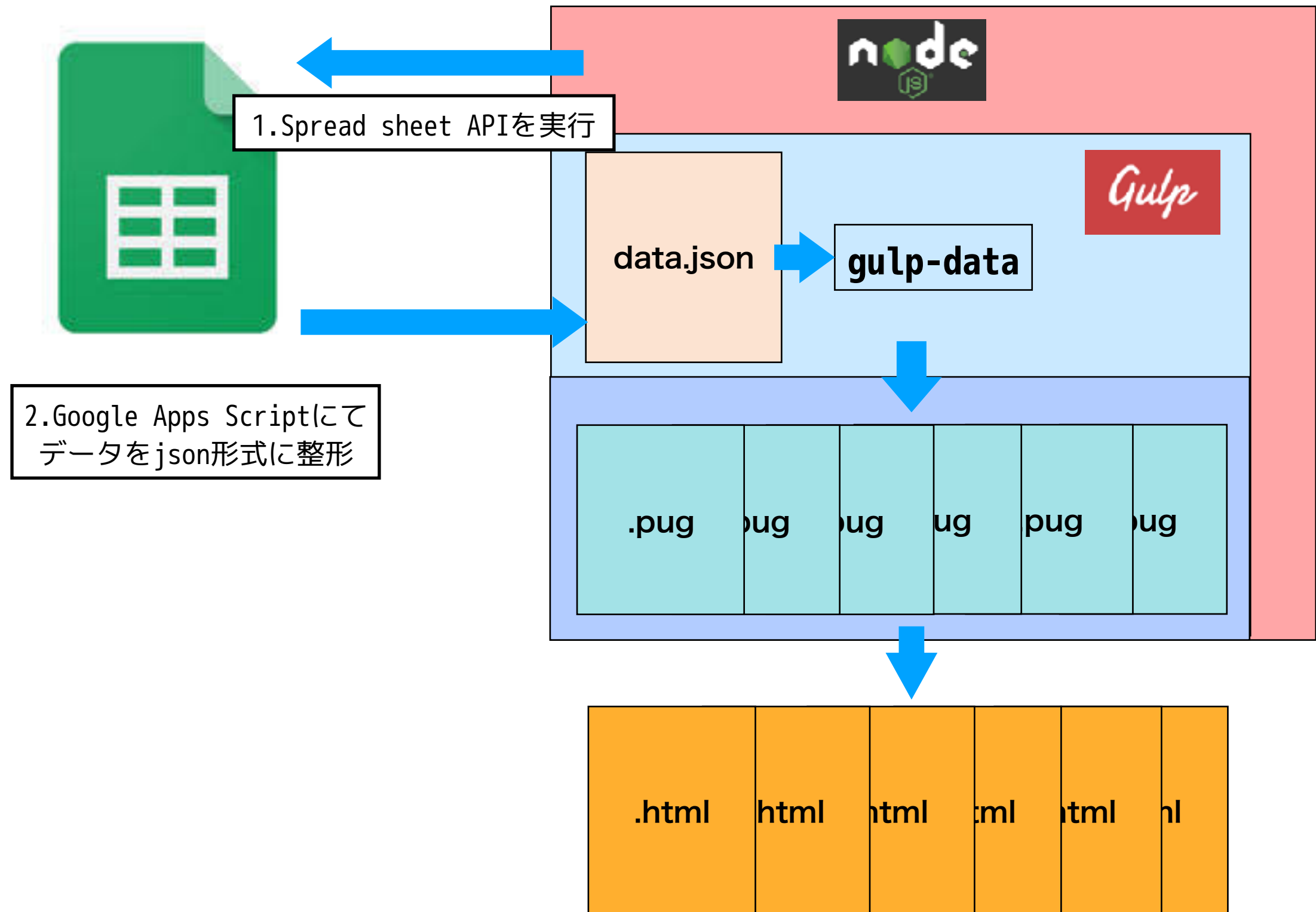
## Google Spreadsheet

- ・オンラインで編集可能
- ・Google Apps Script  
->json形式に整形可

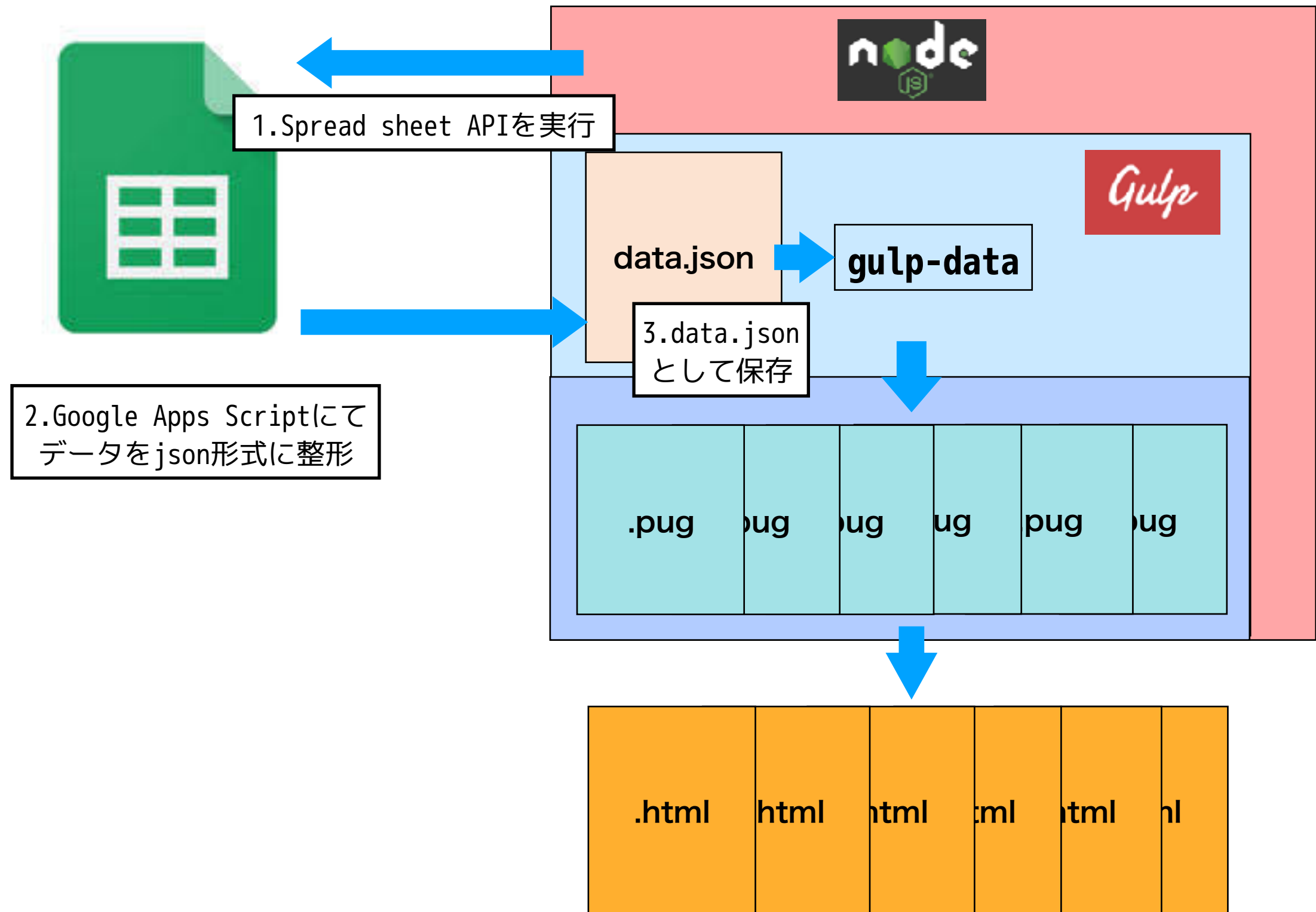
# Pug応用編 データのスプレッドシート管理



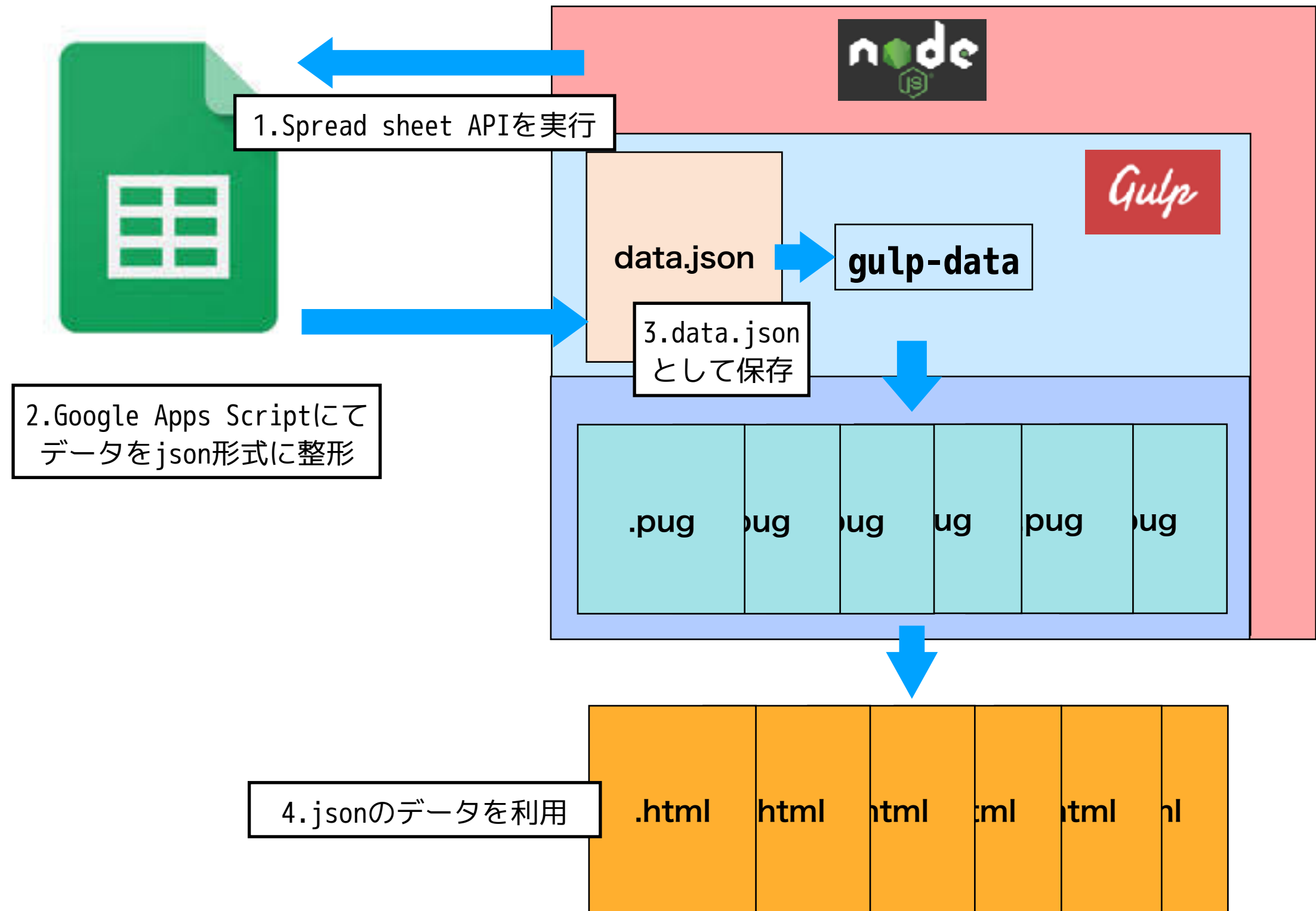
# Pug応用編 データのスプレッドシート管理



# Pug応用編 データのスプレッドシート管理



# Pug応用編 データのスプレッドシート管理



# Pug応用編 データのスプレッドシート管理

---

参考：

**Create a Target Project for the Apps Script Execution API**

<https://developers.google.com/apps-script/guides/rest/quickstart/target-script>

**【GAS】スプレッドシートのデータをJSON出力する関数を  
Execution APIで外部のNode.jsから実行する**

<http://qiita.com/kingpanda/items/8e60a64dc2454f6ae6b5>



# Pugの応用 まとめ

---

- 変数をページの上部にまとめて書くと管理しやすい
- 外部ファイル化することでデータをまとめて管理できる
- スプレッドシートを使えばオンライン上でデータを管理できる。

# 事例の紹介

# 事例の紹介

## 製品統合サイトリニューアル

ページ数	170p	フロント エンド	4名
製作期間	1.5ヶ月	担当	デザイン・フロントエンド

- ・ インクルードファイルを細かく設定し並行して制作
- ・ データはスプレッドシートで管理
  - ・ title/keyword/薬事承認番号・リコメンドバナー

リニューアル後の運用対応

# 事例の紹介

## 金融系サイトリニューアル


ページ数	210p	フロント エンド	4名
製作期間	3ヶ月	担当	フロントエンド・CMS

- パンくずリストをスプレッドシート管理

pugにパスとjsonがわたせれば、ディレクトリの親子関係からパンくずリストが作成できます。たとえディレクトリ構造と合わないケースであってもjsonにタイトルと自身の親が渡せればリストを作成できます。

ホーム > XXについて > XXの取り組み

 index.html

 /about/index.html

 /csr/view/index.html

# 最後に

---

今回はテンプレートエンジンのPugにフォーカスしましたが、Pug以外のテンプレートエンジンもほぼ変わらない機能と使い方ができるはずです。

今後の静的HTMLの作成の際はぜひ取り入れてみてください。

# 質問

---

ご静聴

ありがとうございました。