


# パスの指定

---

ファイル・フォルダ・ディレクトリ・相対パス・絶対パス

作成：斉藤



# Index

---

- ファイル・ディレクトリとは : P3
- パスとは / その種類 : P4
- ディレクトリの種類 : P5
- 絶対パス : P6
- 相対パス : P7
- まとめ : P8

# ファイル・ディレクトリとは

## ファイル

ファイル名とファイルの種類を表す拡張子が付く。

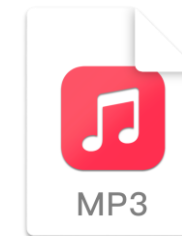
例：IMG\_220519.png    過去問.pdf  
audio.mp3            index.html  
movie.mp4            main.c

## フォルダ(ディレクトリ)

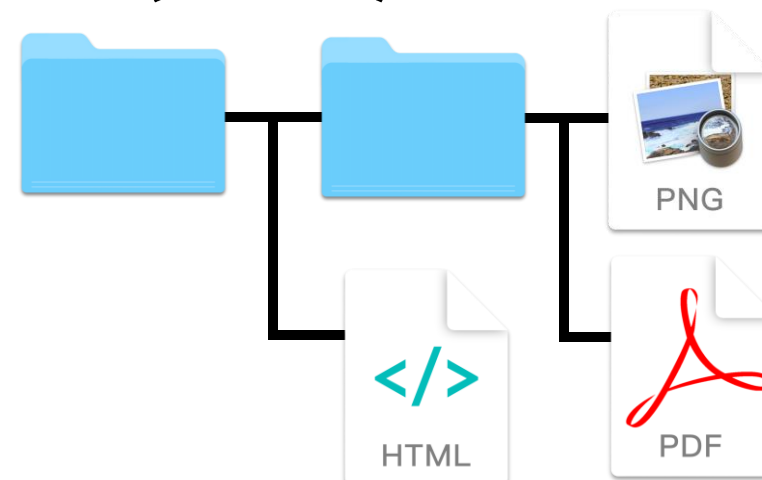
フォルダはディレクトリと呼ぶことがある。  
フォルダの中にはファイルを入れることができる。  
フォルダ名が付く。拡張子はない。  
そのディレクトリの層一帯を階層と呼ぶことも多い。

例：Images            Scripts  
Users                Program Files(x86)

## ファイル



## フォルダ(ディレクトリ)



# ディレクトリの種類

## ルートディレクトリ

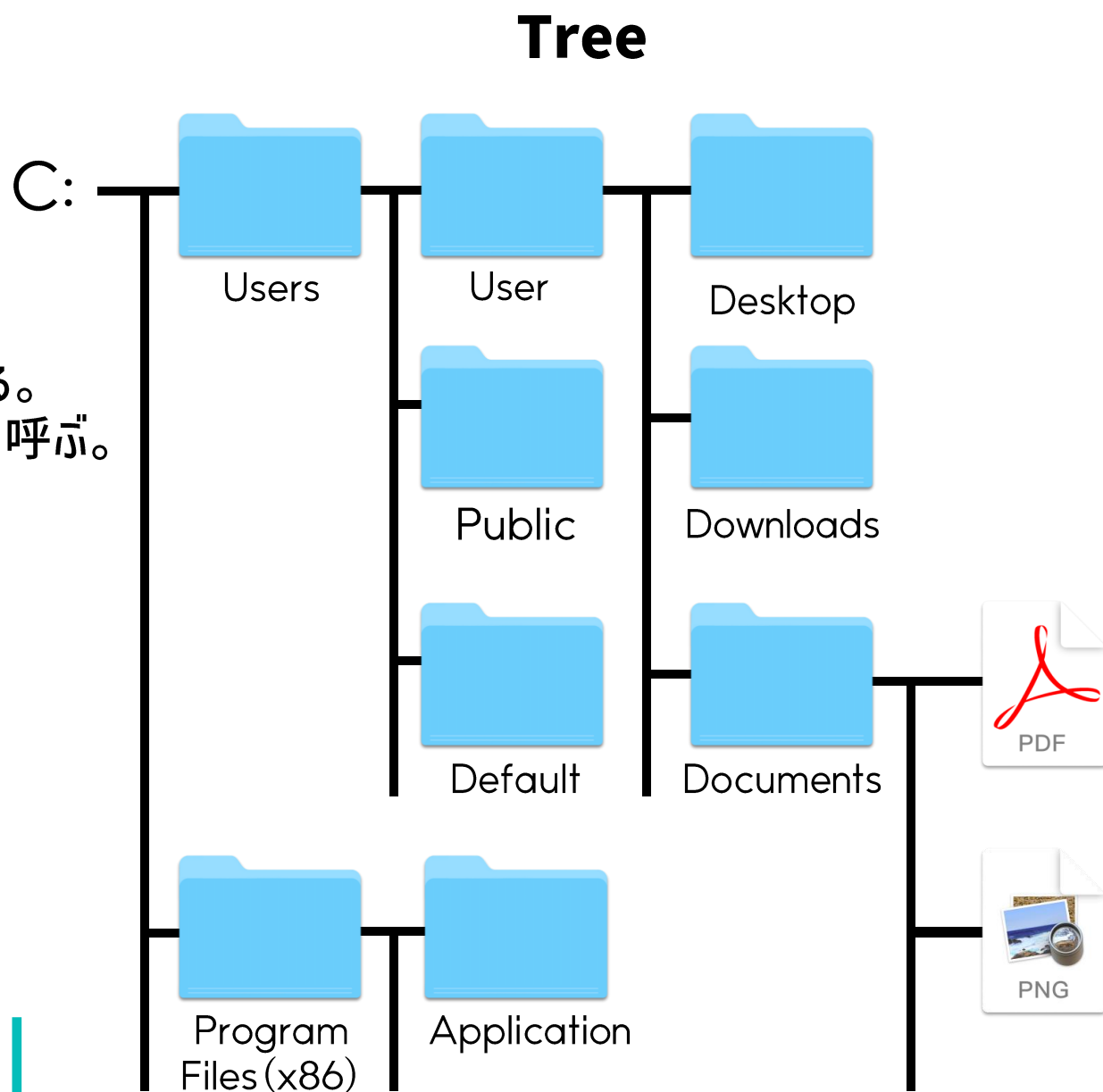
最上位のディレクトリのこと。(= C: )  
すべてのファイルやディレクトリは  
ルートディレクトリを根 (root) とする  
木構造のディレクトリ階層のいずれかに収まっている。  
→この構造のことをツリー(Tree)または階層構造と呼ぶ。  
絶対パスでファイルやディレクトリ指定する際の  
基準の位置。

## カレントディレクトリ

現在位置として示しているディレクトリのこと。  
相対パスでファイルやディレクトリ指定する際の  
基準の位置。

## 親ディレクトリ

階層構造で表現すると1つ上の  
ディレクトリのことを指す。  
カレントディレクトリ ≡ 親ディレクトリ



# パスとは / 種類

## パスとは

ファイルやフォルダがどこにあるのかというような住所のようなもの。

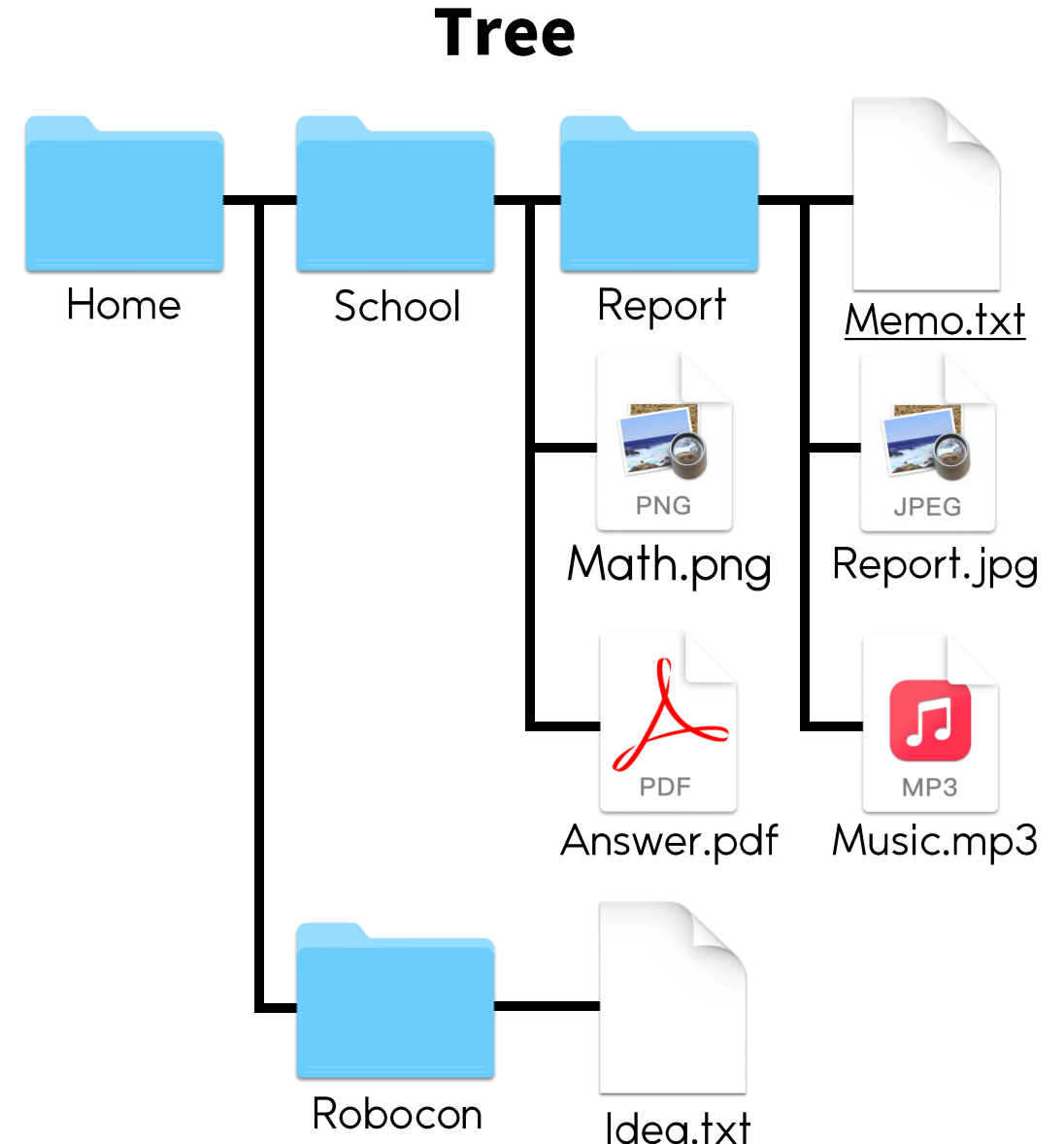
右図を参考にMemo.txtのパス書くと…

Home/School/Report/Memo.txt

このように、上の階層から順に左から書いていき、スラッシュ(“/”)で区切って書く。

※バックスラッシュ(“\”)や円マーク(“¥”)で区切ることもある。

パスの指定方法には、絶対パス・相対パスの2種類がある。



# 絶対パス

## 絶対パス

パスの書き方のうち、  
最も上の階層(ルートディレクトリ)から記述する方法。

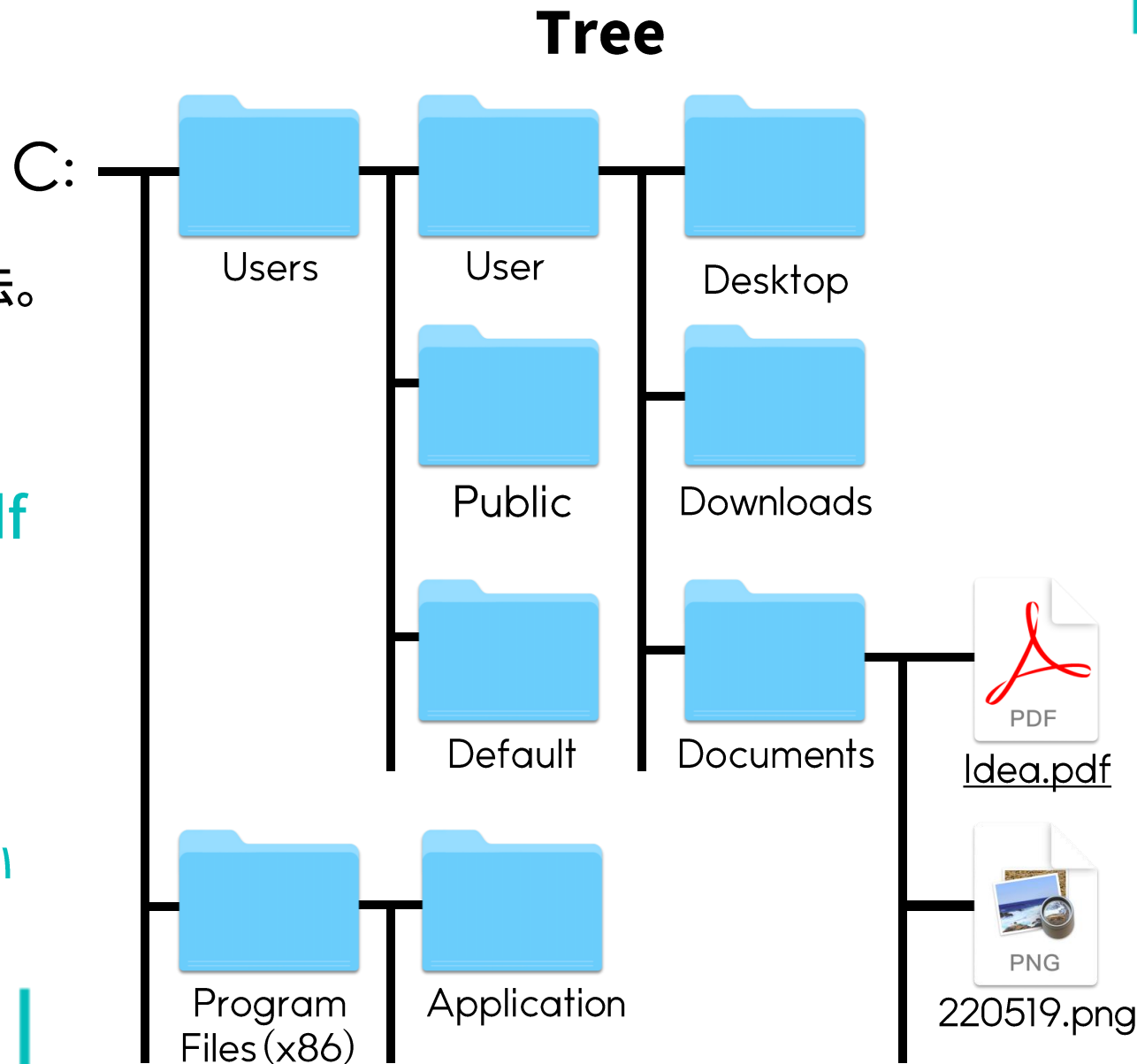
右図を参考にIdea.pdfのパス書くと…

C:/Users/User/Documents/Idea.pdf

目的のファイルの位置が変わらない限り、  
有効である。

メリット： どの階層からでもアクセスできる  
見ただけでどこにあるのかがわかりやすい

デメリット： 長い  
参照するディレクトリに変更があると  
効かなくなる



# 相対パス

## 相対パス

パスの書き方のうち、  
現在の階層(カレントディレクトリ)から記述する方法。

カレントディレクトリをDownloadsとして、  
右図を参考にIdea.pdfのパス書くと…

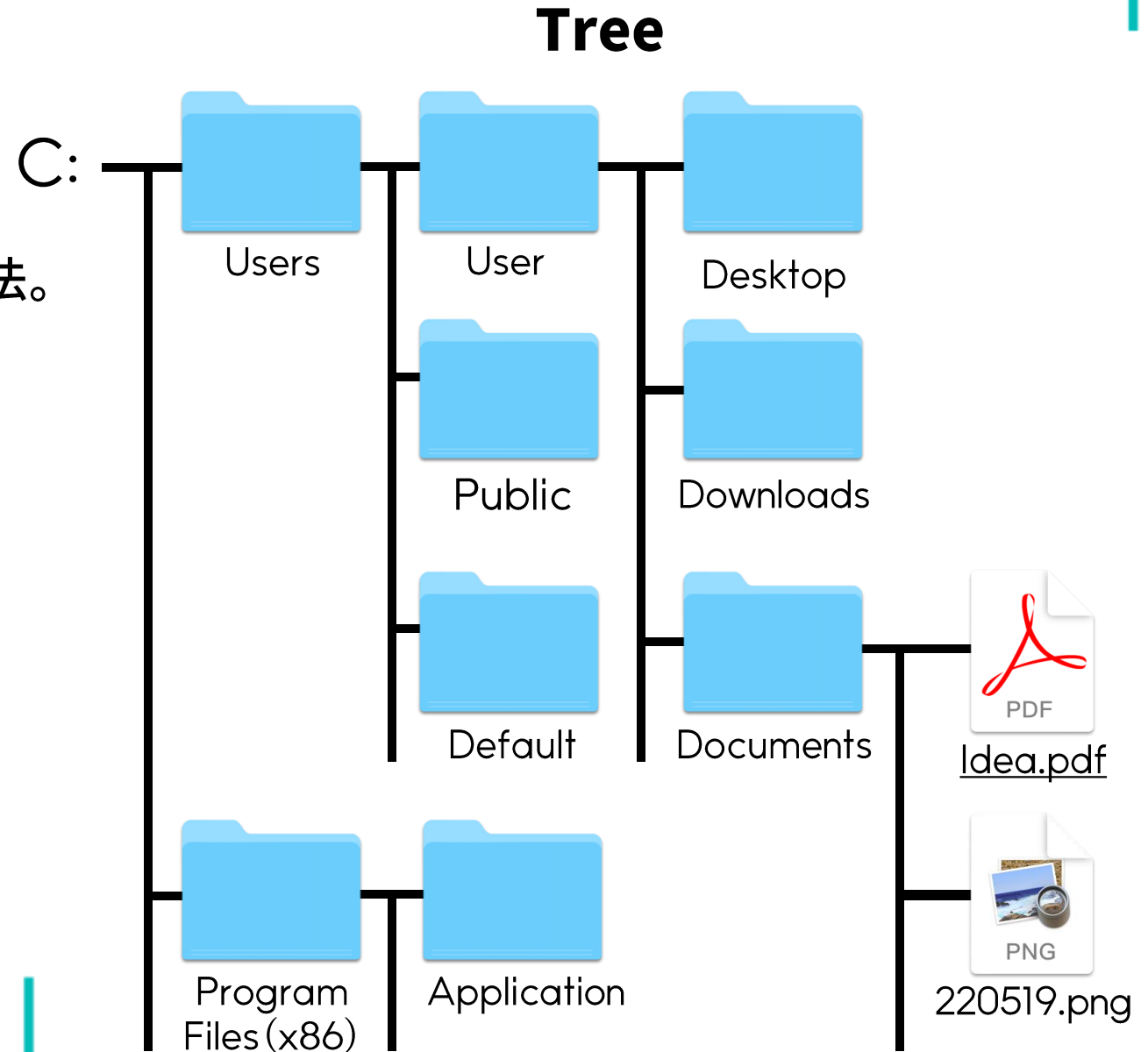
`../Documents/Idea.pdf`

というように、カレントディレクトリ内から見た  
目的のファイルを記述する方法。

メリット： 短い

相手にファイルを渡すときに有効である

デメリット： カレントディレクトリが変わると効かない  
⇒どの階層からでもアクセスできない



# パスの指定まとめ

## 相対パス

現在の階層(カレントディレクトリ)から見た目的のファイルを記述する方法。

メリット：相手にファイルを渡すときに有効である  
デメリット：カレントディレクトリが変わると効かない  
⇒どの階層からでもアクセスできない

## 絶対パス

パスの書き方のうち、最も上の階層(ルートディレクトリ)から記述する方法。

メリット：どの階層からでもアクセスできる  
デメリット：参照するディレクトリに変更があると効かなくなる

