第1章　ファイルの分割と統合

文書の量が多くなると，どこに何が書いてあるのかわかりづらくなり，1 つの TEX ファイルで管理することが難しくなります。そこで，1 つの文書を小分けにして別々で管理し，また別のファイルで統合する，という方法を紹介しておこうと思います。

1.1　ディレクトリとパス

ファイルを分け，統合するにあたって，ディレクトリとパスというものを知っておく必要があります。以下の小節でそれぞれ説明します。知っている人は読み飛ばしてもらって構いません。

1.1.1　ディレクトリ

ディレクトリとは，色々なファイルを入れてあるフォルダのことです。ファイルを人だと考えれば，ディレクトリは家のようなものです。ディレクトリは入れ子（多層構造）にすることができます。例えば，日本というディレクトリに東京都というディレクトリがあり，その中に 23 区のディレクトリがあり，その中に本郷三丁目というファイルがある，といった感じです。このように，層の下に入っていく（狭い範囲になっていく）ことを「下に行く」，逆に広い範囲を含む方へ行くことを「上に行く」ということにします。

1.1.2　パス

パスとは，そのファイルがコンピューターのどこにあるか，という住所のようなものです。パス（住所）を指定することで，目的のファイル（人）を探し出して使うことができるようになります。

パスには「絶対パス」と「相対パス」の 2 種類があります。絶対パスは，一番大元のディレクトリからスタートして指定するものです。住所に例えれば，

宇宙 → 地球 → ユーラシア大陸 → アジア → 日本 → 関東 → 東京 → 文京区

のような感じです。例えば，僕の PC 上でこの TEX マニュアルは，

`C:/Users/User/Document/TeX/TeXManual`

というところにあります。この「Users」が，僕の PC の大元のディレクトリとなっています。

相対パスは，今いるところからの相対位置で指定するものです。住所に例えれば，東京都から東京都に手紙を出すときに，「東京都」の部分を省略して区名から書けば OK，のような感じです。相対パスを指定するとき，同じディレクトリにあるファイルを指定するときはただそのファイル名を書けばよく，今いる（使っているファイルのある）ディレクトリより下のディレクトリのファイルを指定するときは，「ディレクトリ名/ファイル名」とし，今いるディレクトリより上のファイルを指定するときは「../ファイル名」とします。上や下に行った数だけ「/」が書かれることになります

ここで，windows ではパスの区切りが円マーク（¥）やバックスラッシュ（\）となっていますが，TEX でのパスの区切りは全てスラッシュ（/）ですので，注意してください。

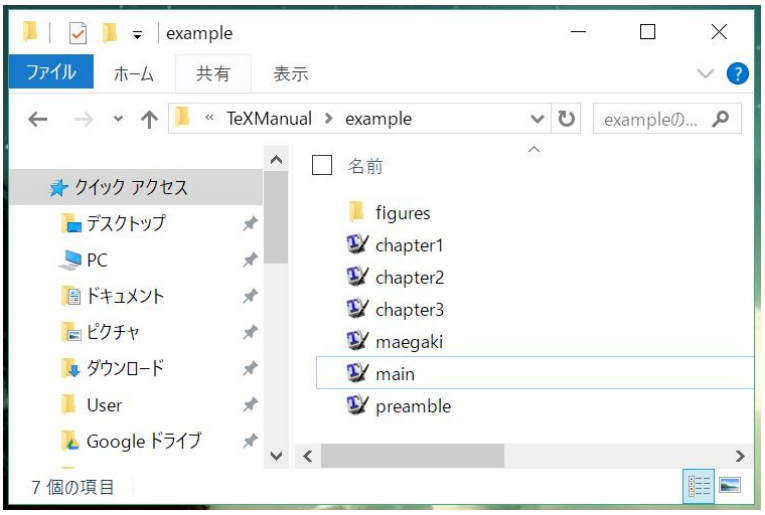
1.2 TEX 標準のコマンド

TEX には，\input{.tex ファイル}と\include{.tex ファイル} という 2 つのコマンドが用意されています。この 2 つはともに，元の.tex ファイルのそのコマンドを，指定された.tex ファイルの内容で置き換える，というものとなっています。違いとしては，\include は改ページあり，\input は改ページ無し，となっています。ちなみに，ファイル名の.tex は省略できます。

1.3 具体例

色々解説してきましたが，実際どのようにするのか見ないとかなりわかりづらいと思うので具体例を示します。

前書き，目次，第 1〜3 章から構成された 1 つの文書を異なるファイルに分割し，統合することを考えます。まずはフォルダを 1 つ作って，前書き（maegaki.tex），第 1〜3 章（chapter1〜3.tex）と，文書に関連するファイルを全てその中に入れます（ここではフォルダ名を example とします）。図があるときはまとめて fgures フォルダにでも入れておきましょう。また，プリアンブル（\documentclass と\begin{document}の間）も分けて別の.tex ファイルに書きます（preamble.tex とします）。最後に統合用の.tex ファイル（main.tex）も同じ所に置いて準備完了です。いま，example フォルダの中身は以下のようになっています。



さて，次は main.tex の中身を見てみましょう。\tableofcontents は目次を出力するコマンドであることに注意してください。

```
\documentclass[dvipdfmx, a4paper]{jsarticle}
\include{preamble}
\begin{document}
\tableofcontents
\include{maegaki}
\include{chapter1}
\include{chapter2}
\include{chapter3}
\end{document}
```

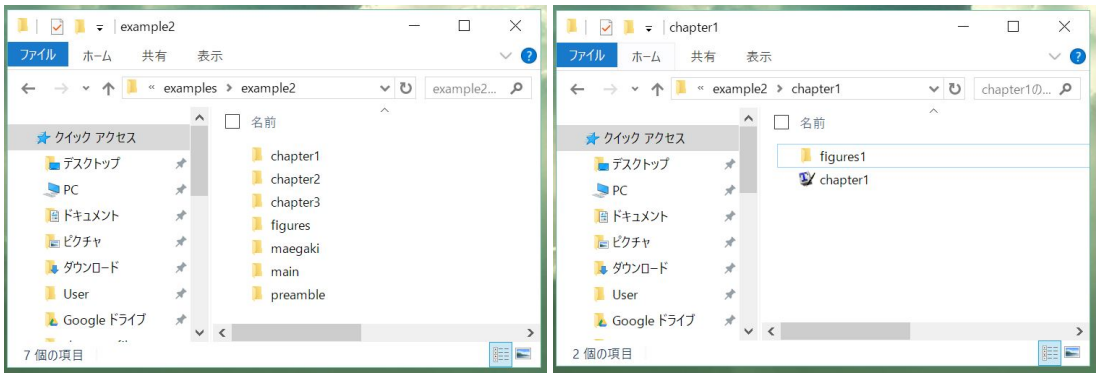
main では分割したファイルを\include を使って統合しています（\include は改ページを伴うので，改ページしたくないときは\input を使ってください）。これで main を実行すれば，これらを全部一緒の.tex ファイルに書いたときと同じ出力が得られます。

なお，前節で説明した通り，\include や\input は読み込んだ.tex ファイルをそこに書き写すだけなので，読み込むファイル（preamble，maegaki，chapter1〜3）には\documentclass〜\begin{document}，\end{document} は書かないでください。

この文書を書いている途中は，chapter3 が用意されていなかったりすると思います。また逆に，chapter2 を書いているときは，目次や chapter1 は不要になるかもしれません。そのときは，\include コマンドの前に%をつけてコメントアウトしてください。

1.4 import パッケージの利用

上では.tex ファイルを全て同じフォルダに入れていますが，図が多かったりするときは章ごとにフォルダを作り，その中に図のフォルダを作ることもあるかもしれません。そのとき，フォルダの中身は例えば以下のようになるでしょう。



このとき，例えば chapter1 を main で\include するには，chapter1.tex のパスを指定しなければなりません。このパスは絶対パスでも相対パスでもいいですが，main からの相対パスで指定する方がよいと思います。これは，絶対パスで指定すると，後々 main のフォルダごと移動するときに全てのパスの指定を書き直さなければならなくなるからです。よって，main.tex では，

```
\include{../chapter1/chapter1.tex}
```

のようにして読み込む形になります（「../」は 1 つ上のディレクトリに移動することを表します）。

しかし，もし chapter1.tex の中で\includegraphics を使って図を読み込んでいるなど，他のファイルを読み込んでいる場合，それがそのまま main.tex に書き込まれる形になるので，chapter1.tex で書いた

\includegraphics の相対パスがずれてしまい，結果図のファイルが見つからずエラーしてしまいます。

このとき，\include で chapter1 を読み込んでいる場合，よくわからないエラーが出ます（多分解決不可です）。

もし\input で読み込んでいる場合，「!LaTeXerror file not found」的なエラーが出ます。こちらは一応解決可能で，chapter1.tex 内で他のファイルを読み込んでいるところ（\include，\input，\includegraphics を使っているところ）のパスの指定を main.tex からの相対パスに変更すれば解決できます。しかしこの方法ではわかりづらい上，main を移動したりファイル名を変更したりしたときに色々なところのパスをいじらなければならず非効率的です。

よって，TEX 標準のコマンドだけでは，フォルダ分けをしてファイルを管理する，つまり\input 等の入れ子を扱うのが非常に面倒です。

そこで，import という便利なパッケージを使います。まずは\usepackage{import}とプリアンブルに書きます。この import では，

```
\import{絶対パス}{ファイル名.tex}
\subimport{相対パス}{ファイル名.tex}
```

という 2 つのコマンドが定義されています。この 2 つのコマンドで chapter1.tex を読み込むと，chapter1.tex 内での\input 等は，chapter1.tex から見たパスとして扱われます。したがって，main から見た相対パスを考えてパスを書く，といった面倒なことはなくなります。

また，これらのコマンドにはそれぞれ，\inputfrom，\subinputfrom という別名が用意されています。なお，\import と \subimport の 2 つを同じファイル内で使うとエラーするので気を付けてください。

この import パッケージを使えば，main.tex の中身は以下のようになります。

```
\documentclass[dvipdfmx, a4paper]{jsarticle}
\input{../preamble/preamble}
\begin{document}
\tableofcontents
\subimport{../maegaki/}{maegaki}
\subimport{../chapter1/}{chapter1}
\subimport{../chapter2/}{chapter2}
\subimport{../chapter3/}{chapter3}
\end{document}
```

これで，例えば chapter1.tex 内で

```
\includegraphics{figures/fig1.pdf}
```

のように，そこからの相対パスで図を読み込んでもちゃんとファイルを見つけてくれます。

この import パッケージはファイルの分割時には非常に便利なので，長い文書を書くときは是非使ってみて下さい。

1.5 分割したファイルを実行する

ここまでで，ファイルを分割してそれをまとめることはできるようになりました。ただ，これだけだとちょっと不便です。例えば，ファイルの一部を書き換え，それを出力して修正するというとき，その実行は main.tex を用いて行わなければなりません。このとき，他のファイルごと実行すると実行時間がかなりかかりますが，他のファイルを読み込んでいる\input などをコメントアウトするのも面倒です。そこで，分割したファイルを実行するやり方を紹介しようと思います。

1.5.1 docmute パッケージ

docmute パッケージを用いれば，\input と\include されるファイルの\begin{document}と\end{document}の外を無視し，その中身だけを読み込むことができます。まずはプリアンブル（preamble.tex）に

```
\usepackage{docmute}
```

と書いてください。ただ，この docmute パッケージはインストールされていない可能性もあります。その場合は，第**??**章**??**節を見て自分でインストールしてもよいですが，後の standalone パッケージがほぼ docmute の上位互換となっているので，そちらを使った方がよいと思います。とりあえずこれはシンプルなパッケージなので，わかりやすい説明のために取り上げておきます。

使い方を説明します。main.tex の中身が以下のようになっているとします。

main.tex

```
\documentclass[dvipdfmx, a4paper]{jsarticle}
\input{../preamble/preamble}
\begin{document}
\include{../chapter1/chapter1}
\include{../chapter2/chapter2}
\include{../chapter3/chapter3}
\end{document}
```

また，chapter1.tex の中身が以下のようになっているとします。

chapter1.tex

```
\documentclass[dvipdfmx, a4paper]{jsarticle}
\input{../preamble/preamble}
\begin{document}
\chapter{はじめに}
はじめに〜〜〜〜〜。

\end{document}
```

まずは chapter1.tex のみを実行し，内容を確認します。内容が OK になったら，main.tex を実行します。そうすると，\begin{document}と\end{document}の間だけが main で読み込まれるので，ファイルの統合もうまくいきます。

ここで注意ですが，プリアンブル部分は無視されるので，main.tex と chapter1.tex のプリアンブルが異なっているとそれぞれの実行結果が変わる可能性があります。できるだけ同じプリアンブルを使うようにしてください。また，\include と\input は使えますが，\import や\subimport は使えないので注意してください。

1.5.2 subfiles パッケージ

subfiles パッケージを使えば，chapter1.tex 等において親の main.tex のプリアンブルを引き継いで使うことができます。まずはプリアンブル（preamble.tex）に

```
\usepackage{subfiles}
```

と書いてください。

使い方ですが，chapter1.tex 等のドキュメントクラスに subfiles を指定し，オプションで main.tex のパスを指定してください。また，読み込みは\subfile コマンドで行ってください。具体的には，以下のようにするということです。

main.tex
```
\documentclass[dvipdfmx, a4paper]{jsarticle}
\input{../preamble/preamble}
\begin{document}
\include{../chapter1/chapter1}
\include{../chapter2/chapter2}
\include{../chapter3/chapter3}
\end{document}
```

chapter1.tex
```
\documentclass[../main/main.tex]{subfiles}
\begin{document}
\chapter{はじめに}
はじめに〜〜〜〜〜。

\end{document}
```

こうすることによって，子の chapter1.tex で main.tex のプリアンブルを読み込んで使うことができます。プリアンブルを共有できるので，いちいち子のファイルでプリアンブルを書いたり読み込んだりする必要がなくて便利です。

ただ，\import や\subimport に対応するコマンドは用意されていませんので注意してください（僕が試したときは，chapter1.tex 等で\input を使うとテキストのみ読み込まれ，tcolorbox や tikz が表示されないという形になりました）。

なお，この小節は「分割した LaTeX ファイルを subfiles を使ってコンパイルする」を参考にしました。

1.5.3 standalone パッケージ

standalone パッケージを使えば，子のプリアンブルを親に共有することができます。まずは，subpreambles オプションを追加して，

```
\usepackage[subpreambles=true]{standalone}
```

としてください。

使い方ですが，main.tex での読み込みは\input や\subimport で行えます。main.tex の中身は上と同じとしますので，省略します。子のファイルについてですが，これは以下のようにします。

chapter1.tex
```
\documentclass[class=jsarticle, crop=false, preview=false, dvipdfmx,
    a4paper]{standalone}
\input{../preamble/preamble}
\newcommand{\kodomo}{子供}
\begin{document}
\chapter{はじめに}
はじめに〜〜〜〜〜。
\kodomo

\end{document}
```

ドキュメントクラスには standalone を指定し，オプションで class=jsarticle 等とします。jsarticle 等につけるオプション（dvipdfmx や a4paper 等）も普通につけることができます。crop=flase や preview=false ですが，僕はよくわからないまま使っているので是非自分で調べてみて下さい。

上の例では，chapter1.tex のプリアンブルで\kodomo というコマンドを定義し，文章中でこれを使っています。docmute や subfile ではこれは main.tex には読み込まれないので，main.tex を実行するとエラーしてしまいます。しかし，standalone を使えば，子のプリアンブルが親にも共有されるので，main.tex を実

行してもエラーしません。他の子のファイルと共有しないようなプリアンブルは，共有のプリアンブルでなく子のそれに書いてもよいでしょう。

standalone パッケージでは，\input や\include に加え\import や\subimport も使えるので，これらを使いたい人は standalone パッケージを使ってみて下さい。

ここで注意ですが，子のプリアンブルが親のプリアンブルに共有されるので，preamble.tex の相対パスが main.tex からと chapter1.tex からで異なる場合はエラーします。絶対パスで指定してもいいですが，特に子のプリアンブルを親に共有しなくてもいい場合は，\usepackage{standalone}のように，subpreambles オプションを外してください。

この standalone パッケージは元々 tikz という描画パッケージを読み込むときに便利なものです。僕はその用途で使っているわけではないのであまりそちらには詳しくありません。そちらの用途で使うとき用のオプションやコマンドが用意されているので，是非自分で調べてみて下さい。

この小節は，「Multi-file LaTeX projects」を参考にしました。

※エラーのメモ

\usepackage[subpreambles=true]{standalone}としているとき，コンパイル時に.sta ファイルが生成される。この.sta ファイルには子のプリアンブルが書き込まれ，それが main で読み込まれるという仕様になっているようだ。ただ，この.sta ファイル，コンパイルエラーが起きたときに\endstandalonepreambles というコマンドを書き込まずに閉じてしまい，その次のコンパイルから「l543. }」で始まるエラーが毎回起こるようになる。.sta ファイルを削除するか\endstandalonepreambles を.sta ファイルの最後に書き足せばエラーはなくなるが，これではかなり面倒である。当面，import が使えるパッケージとしての有用性を発揮してもらうだけになりそうだ。

上の全てのパッケージを通して，子の chapter1.tex 等で使った他ファイルへの参照となる\ref は「??」等となりうまく表示されません。main においてはちゃんと表示されるので基本的にはそこまで気にしなくて大丈夫だと思いますが，子の PDF をそのまま使う可能性がある場合は注意してください。