Latex Resources for Writing Programming Books

劉邦鋒
臺灣大學資訊工程學系

June 20, 2020

Chapter 1

Examples

programmingbook is a package that I use to write programming books. It requires listing package and you can use it to write books in Chinese (with CJK). The latex source will look like this.

```
\documentclass[11pt,twoside]{book}

\usepackage{CJK}
\usepackage{programmingbook}

\begin{document}
\begin{CJK}{UTF8}{bsmi}
% your book here.
\end{CJK}
\end{document}
```

1.1 Program Listing

1.1.1 List the entire program

Use programlisting to list a program. The first argument is the file name, and the second argument is the caption.

```
\programlisting{scan-print.c}{從鍵盤讀入再顯示}
```

The previous code will produce the following.

範例 1.1: (scan-print.c) **從鍵盤讀入再顯示**

```
1 #include <stdio.h>
```

```
2 /* main */
3 main()
4 {
5   int i;
6   scanf("%d", &i);
7   printf("%d\n", i);
8 }
9 /* mainend */
```

Note that you can refer to a program listing by `ex:filename`. For example you can refer to the previous example by 範例~`\ref{ex:scanf-print.c}`, which will produce 範例 1.1.

1.1.2 List a part of a program

Use `programlistingrange` to list a part of a program. The first argument is the file name, and the second argument is the caption. The third and the fourth arguments specify the range (within comments) to print. For example you can print the main program of 範例 1.1 using the following.

```
\programlistingrange{scan-print.c}{從鍵盤讀入再顯示}{main}{mainend}
```

The previous code will produce the following.

程式片段 1.2: (`scan-print.c`) 從鍵盤讀入再顯示

```
3 main()
4 {
5   int i;
6   scanf("%d", &i);
7   printf("%d\n", i);
8 }
```

You can refer to a program segment listing by `ex:filename-start`. The `start` is the starting point (the thid argument) of your `programlistingrange` command. For example you can refer to the previous example by 程式片段~`\ref{ex:scan-print.c-main}`, which will produce 程式片段 1.2.

1.2 Input/Output Listing

Use `programinput` and `programoutput` to list program input and output. Note that the package uses `file.in` and `file.out` as the default file names to print if given the name `file`.

```
\programinput{scan-print}
\programoutput{scan-print}
```

The previous code will produce the following.

輸入

```
1 100
```

輸出

```
1 100
```

1.3 Headers and Prototype

1.3.1 Fucntion Prototype

Use `prototype` to list the prototype of a function. The only argument is the file name.

```
\prototype{strlen.h}
```

The previous code will produce the following.

函式原型 1.3: (`strlen.h`)

```
1 int strlen(char *string);
```

You can refer to a prototype by `prototype:filename`. For example you can refer to the previous prototype by 函式原型`~\ref{prototype:strlen.h}`, which will produce 函式原型 1.3.

1.3.2 Header Files

Use `header` to list a header file. The first argument is the file name, and the second argument is the caption.

```
\header{complex.h}{複數定義標頭檔}
```

The previous code will produce the following.

標頭檔 1.4: (complex.h) 複數定義標頭檔

```
1 struct complex {
2   int real;
3   int imag;
4 };
5 typedef struct complex Complex;
```

Note that you can refer to a program listing by `ex:filename`. For example you can refer to the previous example by 標頭檔~`\ref{header:complex.h}`, which will produce 標頭檔 1.4.

1.4 Phrase

1.4.1 Header Files

Use `phrase` to list a phrase. The first argument is the name of the file storing the phrase, and the second argument is the caption. This package will use `.p` for file with phrases.

Note that I use phrase to explain key programming concepts. A phrase is a code segment that are often used in a particular situation, much like a phrase in English.

```
\phrase{for}{{\tt for} 迴圈}
```

The previous code will produce the following.

片語 1.5: for 迴圈

```
1 for (initialization; condition; adjustment)
2   statement;
```

Note that you can refer to a phrase listing by `phr:for`. For example you can refer to the previous phrase by 片語~`\ref{phr:for}`, which will produce 片語 1.5.