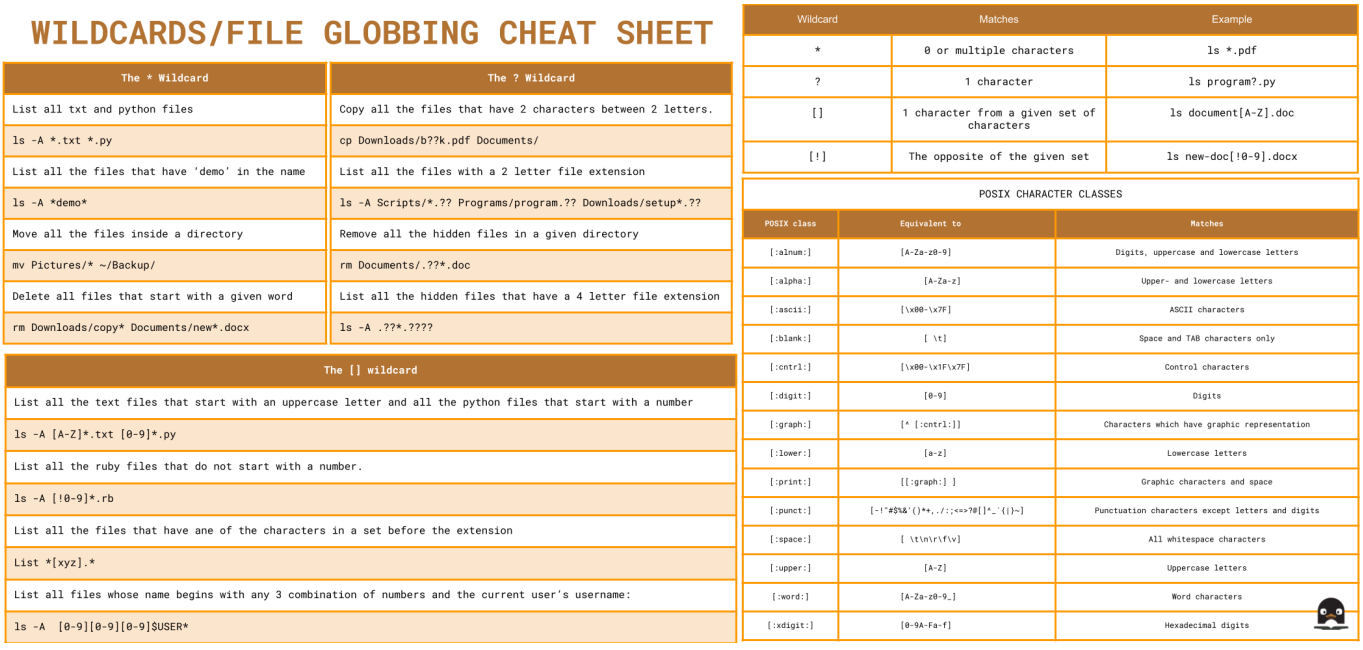
Wildcards

Wildcard represents letters and characters used to specify a file name for searches. They are officially called meta character wildcards. You can use a wildcard to get a long list of all files in the current directory starting with "new."

Their are 3 Wildcards *, ?, and [] each have their own use in the filesystem.

Wildcard	Description
*	Matches zero or more characters in a filename
?	Matches any one character in a filename
[acf]	Matches one of multiple characters in a filename; in this example, a, c, or f
[a-f]	Matches one of a range of characters in a filename; in this example, any character from a through f
[!a-f]	Matches filenames that don't contain a specified range of characters; in this example, filenames that don't contain a through f

This is a cheat sheet that shows many popular uses for the Wildcards.



WILDCARDS/FILE GLOBBING CHEAT SHEET

The * Wildcard	The ? Wildcard
List all txt and python files	Copy all the files that have 2 characters between 2 letters.
ls -A *.txt *.py	cp Downloads/b??k.pdf Documents/
List all the files that have 'demo' in the name	List all the files with a 2 letter file extension
ls -A *demo*	ls -A Scripts/*.?? Programs/program.?? Downloads/setup*.??
Move all the files inside a directory	Remove all the hidden files in a given directory
mv Pictures/* ~/Backup/	rm Documents/.??*.doc
Delete all files that start with a given word	List all the hidden files that have a 4 letter file extension
rm Downloads/copy* Documents/new*.docx	ls -A .??*.????

The [] wildcard
List all the text files that start with an uppercase letter and all the python files that start with a number
ls -A [A-Z]*.txt [0-9]*.py
List all the ruby files that do not start with a number.
ls -A [!0-9]*.rb
List all the files that have ane of the characters in a set before the extension
List *[xyz].*
List all files whose name begins with any 3 combination of numbers and the current user's username:
ls -A [0-9][0-9][0-9]$USER*

Wildcard	Matches	Example
*	0 or multiple characters	ls *.pdf
?	1 character	ls program?.py
[]	1 character from a given set of characters	ls document[A-Z].doc
[!]	The opposite of the given set	ls new-doc[!0-9].docx

POSIX CHARACTER CLASSES

POSIX class	Equivalent to	Matches
[:alnum:]	[A-Za-z0-9]	Digits, uppercase and lowercase letters
[:alpha:]	[A-Za-z]	Upper- and lowercase letters
[:ascii:]	[\x00-\x7F]	ASCII characters
[:blank:]	[\t]	Space and TAB characters only
[:cntrl:]	[\x00-\x1F\x7F]	Control characters
[:digit:]	[0-9]	Digits
[:graph:]	[^ [:cntrl:]]	Characters which have graphic representation
[:lower:]	[a-z]	Lowercase letters
[:print:]	[[:graph:]]	Graphic characters and space
[:punct:]	[-!"#$%&'()*+,./:;<=>?@[]^_`{|}~]	Punctuation characters except letters and digits
[:space:]	[\t\n\r\f\v]	All whitespace characters
[:upper:]	[A-Z]	Uppercase letters
[:word:]	[A-Za-z0-9_]	Word characters
[:xdigit:]	[0-9A-Fa-f]	Hexadecimal digits

Brace Expansion and how to use it

Brace expansion {} is not a wildcard. It is another feature of bash that allows you to generate strings to use with commands.

Using Brace Expansion

- Brace expansion {} is not a wildcard but another feature of bash that allows you to generate arbitrary strings to use with commands.
- For example,
  - To create a whole directory structure in a single command:
    - `mkdir -p music/{jazz,rock}/{mp3files,vidoes,oggfiles}/new{1..3}`
  - To create a N number of files use:
    - `touch website{1..5}.html`
    - `touch file{A..Z}.txt`
    - `touch file{001..10}.py`
    - `touch file{{a..z},{0..10}}.js`
  - Remove multiple files in a single directory
    - `rm -r {dir1,dir2,dir3,file.txt,file.py}`