Fish Shell Scripting Cheatsheet

A comprehensive guide to Fish shell scripting, covering syntax, commands, and best practices for efficient shell scripting.

Fish Shell Basics

Core Syntax

Variable Assignment	`set variable_name value`
Accessing Variables	`$variable_name` or `${variable_name}`
String Concatenation	`set combined "$var1 $var2"`
Command Substitution	`set output (command)`
Comments	`# This is a comment`
Exit Status	`echo $status` (after command execution)

Control Flow

If Statement	`if test condition`
	`  # commands`
	`else if test condition`
	`  # commands`
	`else`
	`  # commands`
	`end`

For Loop	`for i in item1 item2 item3`
	`  # commands using $i`
	`end`

While Loop	`while test condition`
	`  # commands`
	`end`

Switch Statement	`switch $variable`
	`  case value1`
	`  # commands`
	`  case value2`
	`  # commands`
	`  case *`
	`  # default commands`
	`end`

Functions

Function Definition	`function function_name`
	`  # commands`
	`end`

Calling Functions	`function_name arg1 arg2`
Function Arguments	`$argv[1]`, `$argv[2]`, etc.
Return Values	Use `return value` (sets `$status` variable)

Common Commands & Utilities

File and Directory Operations

Create Directory	`mkdir directory_name`
Remove Directory	`rmdir directory_name` (empty directory) `rm -r directory_name` (recursive delete)
Create File	`touch file_name`
Remove File	`rm file_name`
Copy File	`cp source_file destination_file`
Move/Rename File	`mv source_file destination_file`
List Files	`ls`

Text Manipulation

Print to Console	`echo message`
Grep (Search)	`grep pattern file_name`
Sed (Stream Editor)	`sed 's/old/new/g' file_name`
Awk (Pattern Scanning)	`awk '{print $1}' file_name`
String Length	`string length $variable`

Process Management

Run Command in Background	`command &`
List Running Processes	`jobs` or `ps`
Kill Process	`kill process_id`
Foreground Process	`fg job_id`

Advanced Fish Scripting

Error Handling

Check Exit Status	`if test $status -ne 0`
Try…Catch (Simulated)	`command || echo "Error occurred"`
Using `command` to ignore functions	Forces execution of external command instead of shell function with same name. Useful in scripts.

Signals

Trapping Signals	Fish does not directly support `trap`. Use external tools or workarounds.

Input and Output Redirection

Redirect Output to File	`command > file.txt` (overwrite) `command >> file.txt` (append)
Redirect Input from File	`command < file.txt`
Pipe Output	`command1 | command2`
Redirect Standard Error	`command ^ file.txt`

Working with Lists

Creating a List	`set my_list item1 item2 item3`
Accessing List Elements	`echo $my_list[1]` (first element)
List Length	`count $my_list`

Iterating Over a List	`for item in $my_list`
	`  echo $item`
	`end`

Configuration and Customization

Configuration Files

Global Configuration	`~/.config/fish/config.fish`
Functions Directory	`~/.config/fish/functions/` (for custom functions)
Autocompletions Directory	`~/.config/fish/completions/` (for custom autocompletions)

Custom Functions

Creating a Function	`function greet`
	`  echo "Hello, $argv[1]!"`
	`end`

Making it Persistent	Save the function in `~/.config/fish/functions/greet.fish`

Aliases

Creating Aliases (Abbreviations)	`abbr la 'ls -la'`
Making Aliases Persistent	Add `abbr` commands to `~/.config/fish/config.fish`

Autocompletions

Creating Autocompletions	Use `complete` command to define autocompletions. Example: `complete -c mycommand -f -a "option1 option2"`
Custom Completion Files	Store completion definitions in `~/.config/fish/completions/mycommand.fish`