Go Cheatsheet

Go is a programming language that is designed for simplicity, efficiency, and concurrency. Here's a quick overview of its unique features and some code blocks for common tasks.

Variables

Declare variables using the `var` keyword. Go is a statically typed language, but it can infer the data type of a variable from its value.

```go
x := 10
c := 'a'
f := 3.14
```

Functions

Functions in Go are declared using the `func` keyword. They can take parameters and return values.

```go
func add(a int, b int) int {
  return a + b
}
```

Loops

Go supports `for` and `while` loops for iterating over arrays or performing a task a certain number of times.

```go
numbers := []int{1, 2, 3, 4, 5}

for _, number := range numbers {
  fmt.Println(number)
}

i := 0
for i < len(numbers) {
  fmt.Println(numbers[i])
  i++
}
```

Conditionals

Use `if` statements to execute code based on a condition. `else if` and `else` statements can be used to handle multiple conditions.

```go
x := 10

if x > 0 {
  fmt.Println("x is positive")
} else if x < 0 {
  fmt.Println("x is negative")
} else {
  fmt.Println("x is zero")
}
```

File Manipulation

Go provides built-in support for file manipulation, including reading and writing files, and interacting with the file system.

```go
package main

import (
    "fmt"
    "io/ioutil"
    "os"
)

func main() {
    data, err := ioutil.ReadFile("/path/to/file.txt")
    if err != nil {
        fmt.Println("Error reading file")
        return
    }
    fmt.Println(string(data))

    err = ioutil.WriteFile("/path/to/file.txt", []byte("New contents"), 0644)
    if err != nil {
        fmt.Println("Error writing file")
        return
    }

    err = os.Remove("/path/to/file.txt")
    if err != nil {
        fmt.Println("Error deleting file")
        return
    }
    fmt.Println("File deleted successfully")
}
```

Resources

Here are some resources to help you learn more about Go:

- A Tour of Go (Interactive Go tutorial)
- The Go Programming Language Specification (Official Go language specification)
- Effective Go (Go programming best practices)