Chapter 8. Input / Output

2016-10-04

io.Writer Interface

```
type Writer interface {
  Write(p []byte) (n int, err error)
}
```

- `len(p)` 만큼의 바이트를 출력해야 한다.
- `len(p)` 만큼 출력하지 못하면 반드시 `err` 값을 반환해야 한다.
- `n`은 출력한 바이트의 길이이며, `0 <= n <= len(p)` 이어야만 한다.
- `p []byte` 는 수정 되어서는 안된다.

io.Reader Interface

```
type Reader interface {
  Read(p []byte) (n int, err error)
}
```

- `len(p)` 만큼의 바이트를 읽어야 한다.
- 읽은 바이트 수가 0 인 경우에는 `nil`이 아닌 `err` 값을 반환해야 한다.
- 파일의 끝에 도달했을 때에는 `err` 값으로 `EOF` 이나 `nil` 을 반환한다.
- 단, `nil`을 반환하는 경우에는 그 다음번에는 `EOF` 를 반환해야 한다.
- `n`은 읽은 바이트의 길이이며, `0 <= n <= len(p)` 이어야만 한다.
- `p []byte` 는 수정되어서는 안된다.

Example 1

bytes.Buffer's Write Method

```
var b bytes.Buffer
b.Write([]byte("안녕하세요 "))

// `bytes.Buffer`'s `Write` Method
func (b *Buffer) Write(p []byte) (n int, err error) {
    b.lastRead = opInvalid
    m := b.grow(len(p))
    return copy(b.buf[m:], p), nil
}
```

fmt.Fprintf Function

```
fmt.Fprintf(&b, "%s", "Golang!")

// fmt.Fprintf 함수
func Fprintf(w io.Writer, format string, a ...interface{}) (n int, err error)
```

- **Fprintf** 함수는 출력할 **Writer** 를 인자로 받는다.
- 예제 에서는 **fmt.Fprint** 가 더 적절해 보인다.

byte.Buffer's WriteTo Method

```
// byte.Buffer 타입의 WriteTo 메소드
func (b *Buffer) WriteTo(w io.Writer) (n int64, err error)
```

Example 2 - `Curl`

http.Get

```
// http.Response 와 error 반환
r, err := http.Get(os.Args[1])
if err != nil {
    log.Fatalln(err)
}
```

os.Create

```
// os.File 과 error 를 반환
file, err := os.Create(os.Args[2])
if err != nil {
  log.Fatalln(err)
}
// 문제가 없을 시, 함수 종료 와 함께 file 을 닫는다.
defer file.Close()
```

create Writer

```
// multiWriter 를 반환
dest := io.MultiWriter(os.Stdout, file)

type multiWriter struct {
    writers []Writer
}

func (t *multiWriter) Write(p []byte) (n int, err error) {
  // 코드 생략
    return len(p), nil
}
```

- **multiWriter** 는 **Writer** 인터페이스를 갖는 슬라이스로 이루어져 있다.
- **multiWriter** 는 **Write** 함수를 구현해 놓았다.

io.Copy

```
// r.Body(io.ReadCloser)를 os.Stdout 과 os.File 에 복사.
io.Copy(dest, r.Body)

// r.Body 를 닫는다.
if err := r.Body.Close(); err != nil {
  log.Println(err)
}
```

io.Copy Define

```
func Copy(dst Writer, src Reader) (written int64, err error) {
    return copyBuffer(dst, src, nil)
}
```

- io.Copy 함수는 Writer 와 Reader 를 인자로 받는다.
- io.Copy 함수는 src를 dst로 복사한다.

io.ReadCloser

```
type ReadCloser interface {
    Reader
    Closer
}
```

- **ReadCloser** 는 **Reader**, 와 **Closer** 인터페이스를 포함하고 있다.
- 따라서 **Reader**와 **Closer** 인터페이스를 인자로 받는 함수에 전달 할 수 있다.