using vim at work

@danishprakash

danish prakash

software engineer @HackerRank

danishprakash.github.io

what led to this talk?

Mentor

Kindly change your editor by Monday or else it will be difficult for us to work together. See you on Monday!

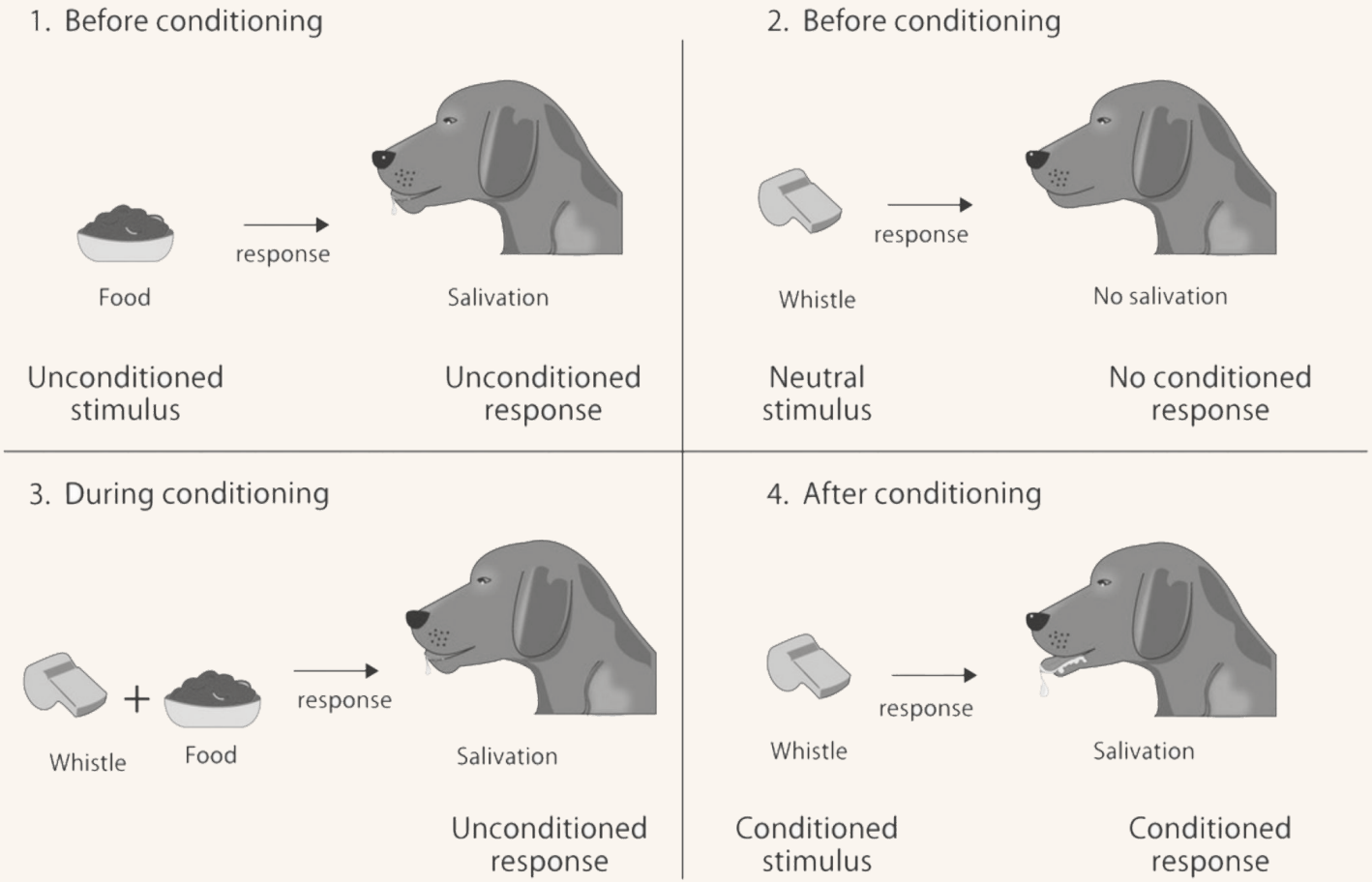
outline

- psychology
- mouse & scroll
- visual cues
- extras
- takeaways

psychology

classical conditioning

an associative learning process that occurs between novel and familiar stimuli



http://changecom.files.wordpress.com/2012/12/classical-conditioning.jpg

mere-exposure effect is a psychological phenomenon by which people tend to develop a preference for things merely because they are familiar with them.

from Wikipedia, the free encyclopedia

learning something new is easier when complemented with familiar elements.

Reder et al
Carnegie Mellon University

mouse & scroll

“but vim promotes a mouse-less workflow!”

mouse is ubiquitous

navigating websites
highlighting text
creating presentations
managing emails
modifying system settings
designing graphics
exploring image galleries

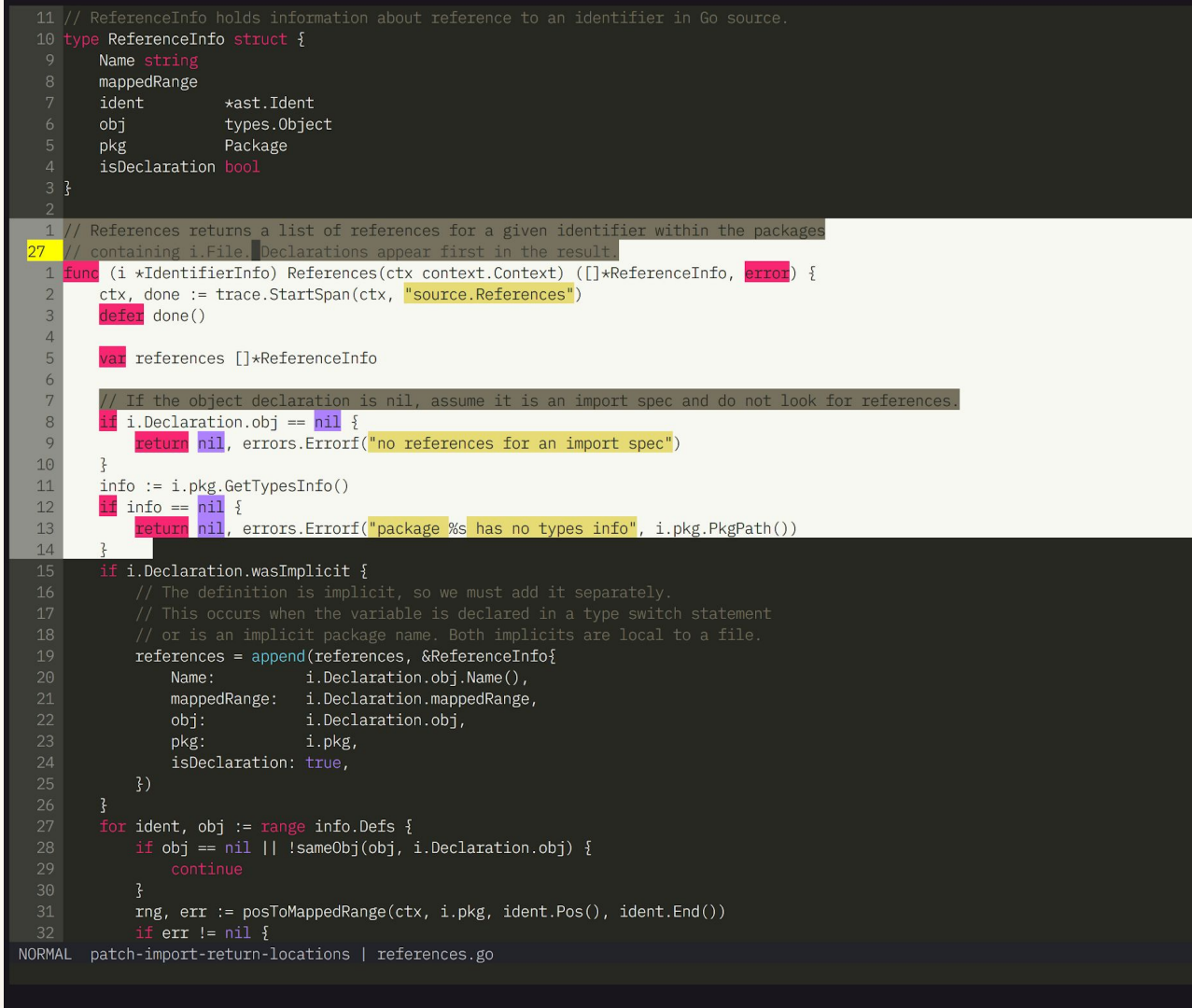
keyboard vs mouse

test subjects report that keyboarding is faster than mousing, whereas stopwatch proves mousing is faster than keyboarding.

Bruce Tognazzini, Apple Interface Design

```
:set mouse-=a
```

mouse selection



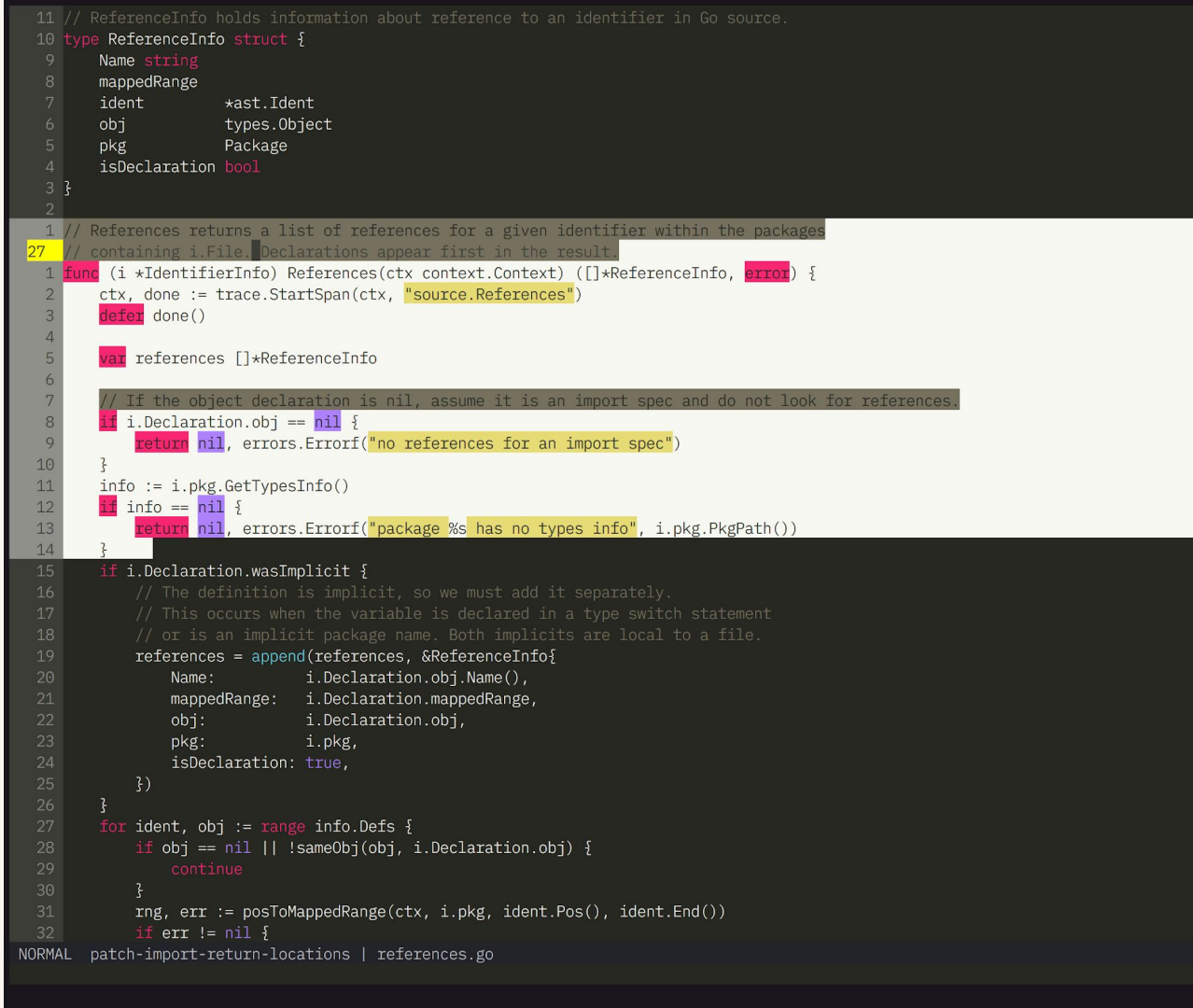
term buffer scroll

```
+github.com/kisielk/gotool v1.0.0/go.mod h1:XhKaO+MFFWcvkIS/tQcRk01m1F5IRFswLeQ+oQHNcck=
+github.com/kr/pretty v0.1.0/go.mod h1:dAy3ld7l9f0ibDNOQOHHMYYIIbhfbHSm3C4ZsoJORNo=
+github.com/kr/pty v1.1.1/go.mod h1:pFQYn66WHrOpPYNljwOMqo10TkYh1fy3cYio2l3bCsQ=
+github.com/kr/text v0.1.0/go.mod h1:4Jbv+DJW3UT/LiOwJeYQe1efqtUx/iVham/4vfdArNI=
+github.com/pmezard/go-difflib v1.0.0 h1:4DBwDE0NGyQoBHbLQYPwSUPoCMWR5BEzIk/f1lZbAQM=
+github.com/pmezard/go-difflib v1.0.0/go.mod h1:iKH77koFhYxTK1pcRnkKkqfTogsbg7gZNVY4sRDYZ/4=
+github.com/rogpeppe/go-internal v1.3.0/go.mod h1:M8bDsm7K2OlrFYOpmOWEs/qY81heoFRclV5y23lUDJ4=
+github.com/sergi/go-diff v1.0.0 h1:Kpca3qRNrduNnOQeazBd0ysaKrUJiIuISHxogkT9RPQ=
+github.com/sergi/go-diff v1.0.0/go.mod h1:0CfEIISq7TuYL3j771MWULgwwjU+GofnZX9QAmXWZgo=
+github.com/stretchr/objx v0.1.0/go.mod h1:HFkY916IF+rwdDfMAkV7OtwuqBVzrE8GR6GFx+wExME=
+github.com/stretchr/testify v1.4.0 h1:2E4SXV/wtOkTonXsotYi4li6zVWxYlZuYNCXe9XRJyk=
+github.com/stretchr/testify v1.4.0/go.mod h1:j7eGeouHqKxXV5pUuKE4zz7dFj8WfuZ+81PSLYec5m4=
 golang.org/x/crypto v0.0.0-20190308221718-c2843e01d9a2/go.mod h1:djNgcEr1/C05ACkg1iLfiJU5Ep61QUkGW8qpdssI0+w=
+golang.org/x/crypto v0.0.0-20190510104115-cbcb75029529/go.mod h1:yigFU9vqHzYiE8UmvKecakEJjdnWj3jj499lnFckfCI=
+golang.org/x/mod v0.0.0-20190513183733-4bf6d317e70e/go.mod h1:mXi4GBBbnImb6dmsKGUJ2LatrhH/nqhxcFungHvyanc=
+golang.org/x/net v0.0.0-20190311183353-d8887717615a/go.mod h1:t9HGtf8HONx5eT2rtn7q6eTqICYqUVnKs3thJo3Qplg=
+golang.org/x/net v0.0.0-20190404232315-eb5bcb51f2a3/go.mod h1:t9HGtf8HONx5eT2rtn7q6eTqICYqUVnKs3thJo3Qplg=
 golang.org/x/net v0.0.0-20190620200207-3b0461eec859 h1:R/3boaszxrf1GEUWTVDzSKVwLmSJpwZ1yqXm8j0v2QI=
 golang.org/x/net v0.0.0-20190620200207-3b0461eec859/go.mod h1:z5CRVTTTmAJ677TzLLGU+0bjPO0LkuOLi4/5GtJWs/s=
 golang.org/x/sync v0.0.0-20190423024810-112230192c58 h1:8gQV6CLnAEikrhgkHFbMAEhagSSnXWGV915qUMm9mrU=
 golang.org/x/sync v0.0.0-20190423024810-112230192c58/go.mod h1:RxMgew5VJxzue5/jJTE5uejpjVlOe/izrB70Jof72aM=
(patch-import-return-locations) tools/gopls $
```

```
    "golang.org/x/tools/go/ast/astutil"
    "golang.org/x/tools/internal/lsp/protocol"
    "golang.org/x/tools/internal/span"
    "golang.org/x/tools/internal/telemetry/trace"
    errors "golang.org/x/xerrors"
)

// IdentifierInfo holds information about an identifier in Go source.
type IdentifierInfo struct {
    Name string
    File ParseGoHandle
    mappedRange

    Type struct {
        mappedRange
        Object types.Object
    }

    Declaration Declaration

    pkg             Package
    ident           *ast.Ident
    wasEmbeddedField bool
    qf              types.Qualifier
```

```
    f, err := view.GetFile(ctx, uri)
    if err != nil {
        return nil, err
    }
    ident, err := source.Identifier(c
    if err != nil {
        return nil, err
    }
    decRange, err := ident.Declaratio
    if err != nil {
        return nil, err
    }
    return []protocol.Location{
        {
            URI:   protocol.NewURI(id
            Range: decRange,
        },
    }, nil
}

func (s *Server) typeDefinition(ctx c
TypeDefinitionParams) ([]protocol.Loc
    uri := span.NewURI(params.TextDoc
    view := s.session.ViewOf(uri)
```

clipboard ambiguity



```
// ReferenceInfo holds information about reference to an identifier in Go source.
type ReferenceInfo struct {
    Name string
    mappedRange
    ident         *ast.Ident
    obj           types.Object
    pkg           Package
    isDeclaration bool
}

// References returns a list of references for a given identifier within the packages
// containing i.File. Declarations appear first in the result.
func (i *IdentifierInfo) References(ctx context.Context) ([]*ReferenceInfo, error) {
    ctx, done := trace.StartSpan(ctx, "source.References")
    defer done()

    var references []*ReferenceInfo

    // If the object declaration is nil, assume it is an import spec and do not look for references.
    if i.Declaration.obj == nil {
        return nil, errors.Errorf("no references for an import spec")
    }
    info := i.pkg.GetTypesInfo()
    if info == nil {
        return nil, errors.Errorf("package %s has no types info", i.pkg.PkgPath())
    }
    if i.Declaration.wasImplicit {
        // The definition is implicit, so we must add it separately.
        // This occurs when the variable is declared in a type switch statement
        // or is an implicit package name. Both implicits are local to a file.
        references = append(references, &ReferenceInfo{
            Name:          i.Declaration.obj.Name(),
            mappedRange:   i.Declaration.mappedRange,
            obj:           i.Declaration.obj,
            pkg:           i.pkg,
            isDeclaration: true,
        })
    }
    for ident, obj := range info.Defs {
        if obj == nil || !sameObj(obj, i.Declaration.obj) {
            continue
        }
        rng, err := posToMappedRange(ctx, i.pkg, ident.Pos(), ident.End())
        if err != nil {
NORMAL  patch-import-return-locations | references.go
```

bottomline

:set mouse=a

even if you are a keyboarder

unexpected surprises

no more of them

pair programming

easy on the person next to you

visual cues

“visual what? and why?”

the human brain can process visual information 60,000 times faster than textual.

3M
Meeting Network

effective use of visuals can decrease learning time, improve comprehension, enhance retrieval, and increase retention.

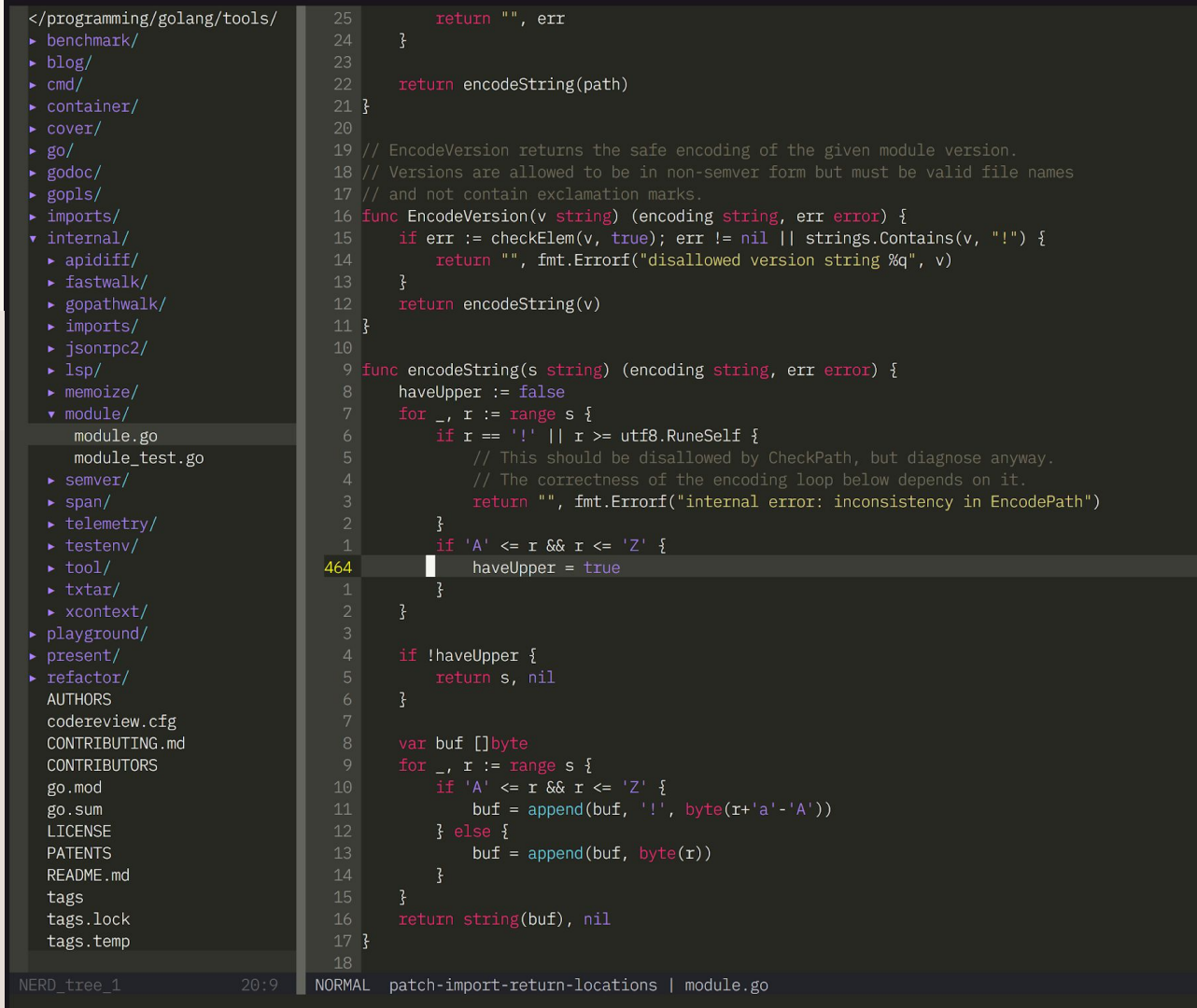
Haig Kouyoumdjian Ph.D.
Psychology Today

familiar stimuli

directory tree

netrw (:Vexplore)

NerdTree

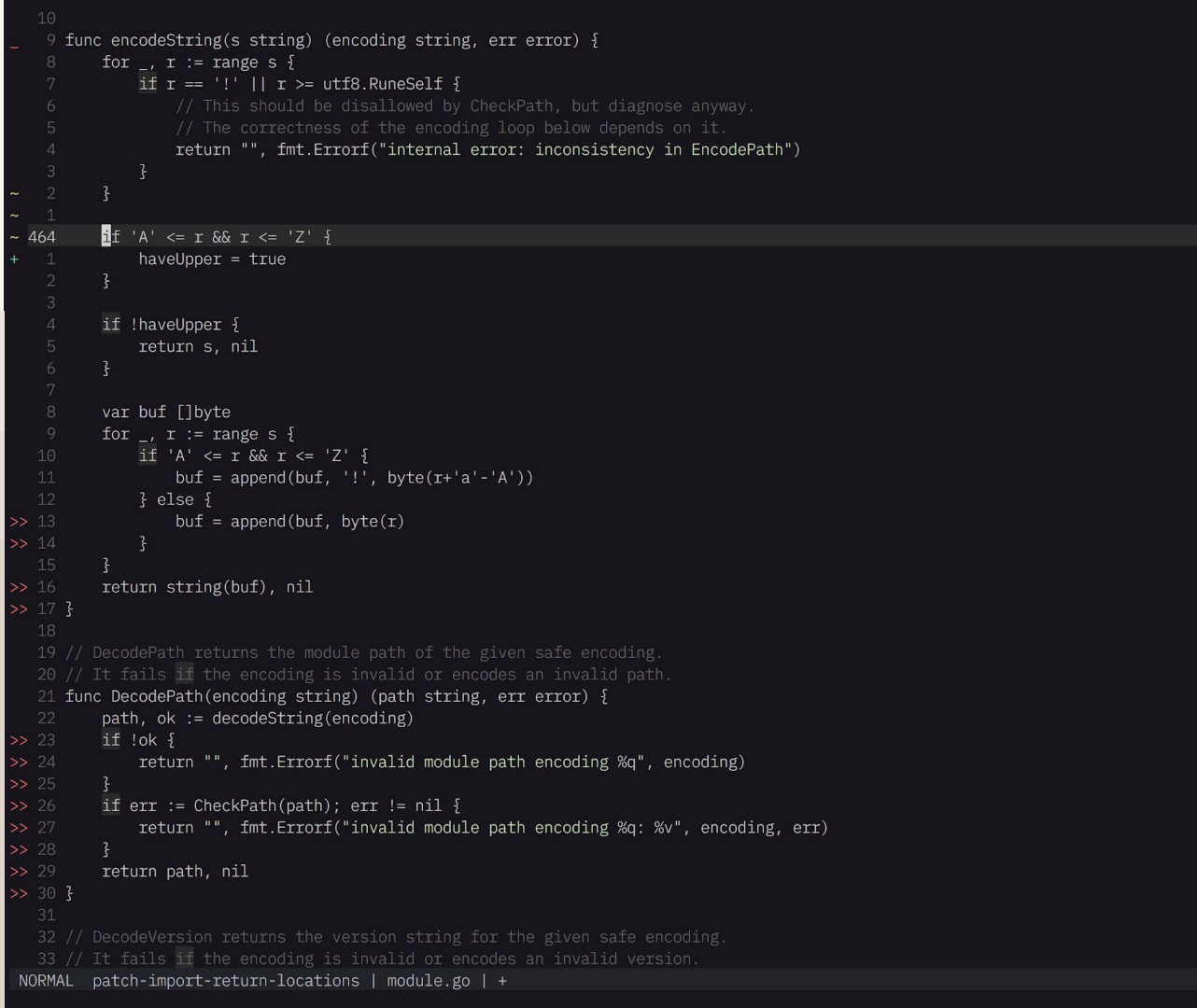


sign column

linter/formatter

version control info

debugger highlights

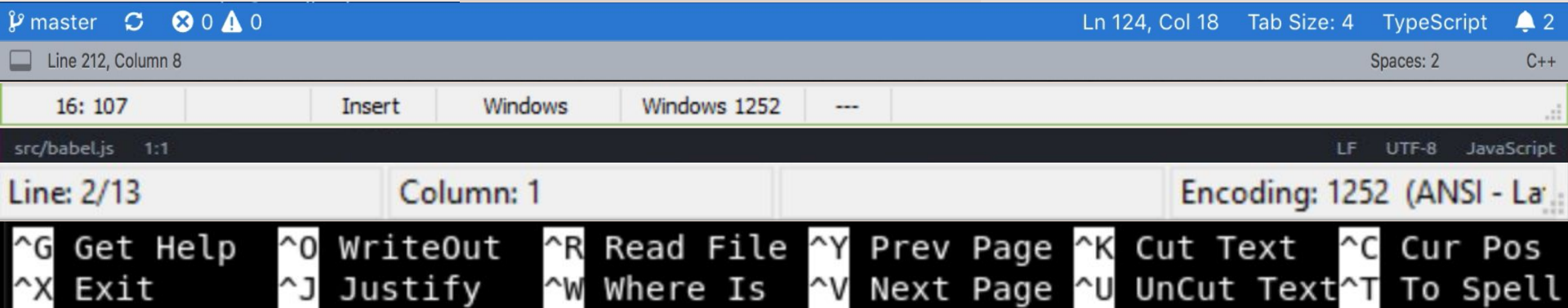


status line

branch info | filename | cursor row/column

language encoding | tabs/spaces | editing mode

linting errors | buffer identifier | filetype



status line

create your own

lightline/airline

```
26
25 // EncodePath returns the safe encoding of the given module path.
24 // It fails if the module path is invalid.
23 func EncodePath(path string) (encoding string, err error) {
22     if err := CheckPath(path); err != nil {
21         return "", err
20     }
19
18     return encodeString(path)
17 }
16
15 // EncodeVersion returns the safe encoding of the given module version.
14 // Versions are allowed to be in non-semver form but must be valid file names
13 // and not contain exclamation marks.
12 func EncodeVersion(v string) (encoding string, err error) {
11     if err := checkElem(v, true); err != nil || strings.Contains(v, "!") {
10         return "", fmt.Errorf("disallowed version string %q", v)
 9     }
 8     return encodeString(v)
 7 }
 6
 5 func encodeString(s string) (encoding string, err error) {
 4     haveUpper := false
 3     for _, r := range s {
 2         if r == '!' || r >= utf8.RuneSelf {
 1             // This should be disallowed by CheckPath, but diagnose anyway.
460             // The correctness of the encoding loop below depends on it.
 1             return "", fmt.Errorf("internal error: inconsistency in EncodePath")
 2         }
 3         if 'A' <= r && r <= 'Z' {
 4             haveUpper = true
 5         }
 6     }
 7
 8     if !haveUpper {
 9         return s, nil
10     }
NORMAL  patch-import-return-locations | module.go | +                    go   85%  460:8
```

extras

tags
ctags, exuberant ctags

git/github integration
vim-fugitive, vim-githubinator

language server protocol
coc.nvim, vim-lsp

sane configs
cursorline, undofile

search
fzf, ctrl-p

effective writing
snippets, vim-commentary, vim-surround

takeaways

use features you are already familiar with

it makes it easier to learn and get accustomed to new information

make the most out of visual cues

our brains are better at processing visuals, use it to your advantage

try not to be repulsive towards new stimuli

whether it’s a new plugin, pattern or a feature altogether

thank you
ありがとう

references

overview of classical conditioning
https://www.verywellmind.com/classical-conditioning-2794859

mere-exposure effect
https://en.wikipedia.org/wiki/Mere-exposure_effect

Building knowledge requires bricks, not sand
Lynne M. Reder, Xiaonan L. Liu, Alexander Keinath, and Vencislav Popov

mouse vs keyboard, AskTog
https://www.asktog.com/TOI/toi06KeyboardVMouse1.html

human brain and visual cues
http://web.archive.org/web/20001102203936/http%3A//3m.com/meetingnetwork/files/meetingguide_pres.pdf

effects of visual learning
https://www.psychologytoday.com/us/blog/get-psyched/201207/learning-through-visuals