

using vim at work

@danishprakash

danish prakash

software engineer @HackerRank

<https://danishpraka.sh/>

what led to this talk?



Mentor

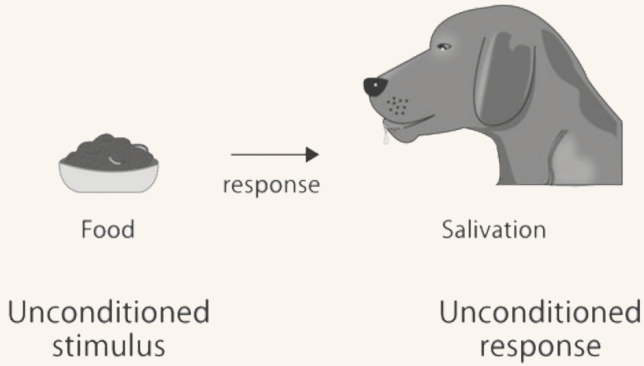
Kindly change your editor by Monday or else it will be difficult for us to work together. See you on Monday!

psychology

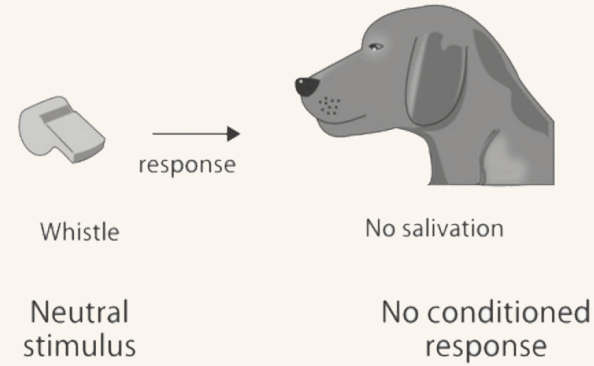
classical conditioning

an associative learning process that occurs
between novel and familiar stimuli

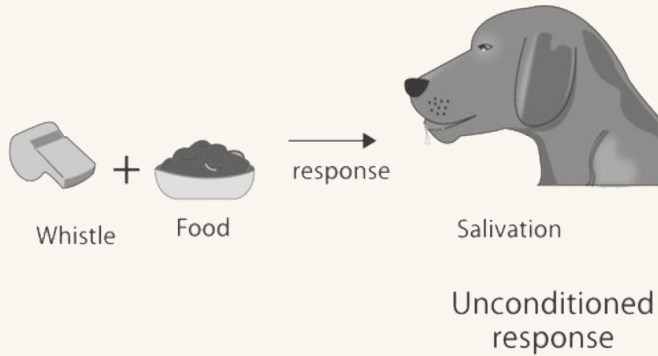
1. Before conditioning



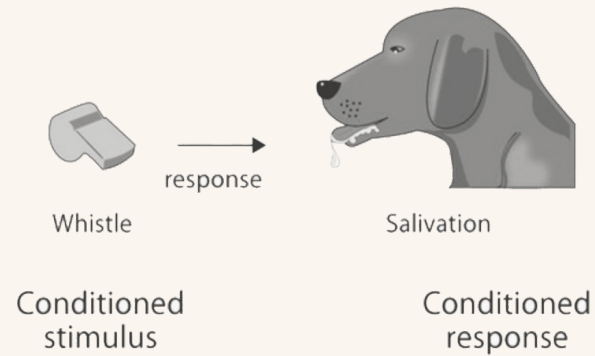
2. Before conditioning



3. During conditioning



4. After conditioning



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 (like this one)

managing emails

modifying system settings (non GNU/Linux systems)

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

```
11 // ReferenceInfo holds information about reference to an identifier in Go source.
10 type ReferenceInfo struct {
9     Name string
8     mappedRange
7     ident      *ast.Ident
6     obj         types.Object
5     pkg         Package
4     isDeclaration bool
3 }
2
1 // References returns a list of references for a given identifier within the packages
27 // containing i.File. Declarations appear first in the result.
1 func (i *IdentifierInfo) References(ctx context.Context) ([]*ReferenceInfo, error) {
2     ctx, done := trace.StartSpan(ctx, "source.References")
3     defer done()
4
5     var references []*ReferenceInfo
6
7     // If the object declaration is nil, assume it is an import spec and do not look for references.
8     if i.Declaration.obj == nil {
9         return nil, errors.Errorf("no references for an import spec")
10    }
11    info := i.pkg.GetTypesInfo()
12    if info == nil {
13        return nil, errors.Errorf("package %s has no types info", i.pkg.PkgPath())
14    }
15    if i.Declaration.wasImplicit {
16        // The definition is implicit, so we must add it separately.
17        // This occurs when the variable is declared in a type switch statement
18        // or is an implicit package name. Both implicits are local to a file.
19        references = append(references, &ReferenceInfo{
20            Name:      i.Declaration.obj.Name(),
21            mappedRange: i.Declaration.mappedRange,
22            obj:       i.Declaration.obj,
23            pkg:       i.pkg,
24            isDeclaration: true,
25        })
26    }
27    for ident, obj := range info.Defs {
28        if obj == nil || !sameObj(obj, i.Declaration.obj) {
29            continue
30        }
31        rng, err := posToMappedRange(ctx, i.pkg, ident.Pos(), ident.End())
32        if err != nil {

```

NORMAL patch-import-return-locations | references.go

term buffer scroll

```
+github.com/kisielk/gotool v1.0.0/go.mod h1:XhKa0aMFFwMcvKIS/tQcRk01m1F5IRfswLeQo+oQhNcck=
+github.com/kr/pretty v0.1.0/go.mod h1:dAy3ld719f0ibDNOQOHMYIIBhfbHSm3C4ZsoJ0RNo=
+github.com/kr/pty v1.1.1/go.mod h1:pFYQYn66WHrOpPYNljw0Mqo10TkYh1fy3cYio2l3bCsQ=
+github.com/kr/text v0.1.0/go.mod h1:4Jbv+DJW3UT/Li0wJeYQe1efqtUx/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:M8bDsm7K20lRfY0pmOWEs/qY81heoFRclV5y23lUDJ4=
+github.com/sergi/go-diff v1.0.0 h1:Kpca3QRNrdunN0QeazBd0ysaKrUjiiuISHxogkT9RPQ=
+github.com/sergi/go-diff v1.0.0/go.mod h1:0CFEIIISq7TuYL3j771MwULgwwjU+GofnZX9QAmXWZgo=
+github.com/stretchr/objx v0.1.0/go.mod h1:HFkY916IF+rwdDfMAKV70twuqBVzrE8GR6GFx+wEXMe=
+github.com/stretchr/testify v1.4.0 h1:2E4SXV/wt0KtonXsotYi4li6zVwxYlZuYNCXe9XRJyk=
+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/C05ACkg1i1fJU5Ep61QUkGW8qpdssI0+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:mXi4GBBBbnImb6dmsKGUJ2LatrhH/nqhxcFungHvyanc=
+golang.org/x/net v0.0.0-20190311183353-d8887717615a/go.mod h1:t9HGtf8HONx5eT2rt7q6eTqICyQUVnKs3thJo3Qp1g=
+golang.org/x/net v0.0.0-20190404232315-eb5bcb51f2a3/go.mod h1:t9HGtf8HONx5eT2rt7q6eTqICyQUVnKs3thJo3Qp1g=
golang.org/x/net v0.0.0-20190620200207-3b0461eec859 h1:R/3boaszxf16EUWTVDzSKVwLmS3pwZ1yqXm8j0v2QI=
golang.org/x/net v0.0.0-20190620200207-3b0461eec859/go.mod h1:z5CRVTTtmAJ677TzLLGU+0bjP00Lku0Li4/5GtJWs/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/jJTE5uejppjVl0e/izrB70Jof72aM=
(patch-import-return-locations) tools/gopls $
```

```
31 "golang.org/x/tools/go/ast/astutil"
30 "golang.org/x/tools/internal/lsp/protocol"
29 "golang.org/x/tools/internal/span"
28 "golang.org/x/tools/internal/telemetry/trace"
27 errors "golang.org/x/xerrors"
26 )
25
24 // IdentifierInfo holds information about an identifier in Go source.
23 type IdentifierInfo struct {
22     Name string
21     File ParseGoHandle
20     mappedRange
19
18     Type struct {
17         mappedRange
16         Object types.Object
15     }
14
13     Declaration Declaration
12
11     pkg          Package
10     ident        *ast.Ident
9     wasEmbeddedField bool
8     qf           types.Qualifier
```

```
24 f, err := view.GetFile(ctx, uri)
23 if err != nil {
22     return nil, err
21 }
20 ident, err := source.Identifier(c
19 if err != nil {
18     return nil, err
17 }
16 decRange, err := ident.Declaratio
15 if err != nil {
14     return nil, err
13 }
12 return []protocol.Location{
11     {
10         URI: protocol.NewURI(id
9         Range: decRange,
8     },
7 }, nil
6 }
5
4 func (s *Server) typeDefinition(ctx c
TypeDefinitionParams) ([]protocol.Loc
3 uri := span.NewURI(params.TextDoc
2 view := s.session.ViewOf(uri)
```

clipboard ambiguity

```
+github.com/kisielk/gotool v1.0.0/go.mod h1:XhKa0oMFFwcVKS/tQcRk0m1F5IRfswLeQo+oQhNcck=
+github.com/kr/pretty v0.1.0/go.mod h1:dAy3ld719f0ibDNOQOHMYIibhfbHSm3C4ZsoJ0RNo=
+github.com/kr/pty v1.1.1/go.mod h1:pFYQYn6WHrOpPYNljwOmQo10TkYh1fy3cYio2l3bCsQ=
+github.com/kr/text v0.1.0/go.mod h1:4Jbv+DJW3UT/Li0wJeyQe1efqtUx/iVham/4vfdArNI=
+github.com/pmezard/go-difflib v1.0.0 h1:4DBwDE0NGyQoBhBLQYPwSUPoCMWR5BEzIk/f1lZbAQM=
+github.com/pmezard/go-difflib v1.0.0/go.mod h1:iKH77koFhYxTK1pcRnkKqfTogsbg7gZNVY4sRDYz/4=
+github.com/rogpeppe/go-internal v1.3.0/go.mod h1:M8bDsm7K20lRfY0pmOWEs/qY81heoFRclV5y23lUDJ4=
+github.com/sergi/go-diff v1.0.0 h1:Kpca3QRNrdUnN0QeazBd0ysaKrUJiiIuISHxogkT9RPQ=
+github.com/sergi/go-diff v1.0.0/go.mod h1:0CFEIIISq7TuYL3j771MwULgwwjU+GofnZX9QAmXWZgo=
+github.com/stretchr/objx v0.1.0/go.mod h1:HFkY916IF+rwDfMAKV70twuqBVzrE8GR6GFx+wEXME=
+github.com/stretchr/testify v1.4.0 h1:2E4SXV/wt0KtonXsotYi4li6zVwxYlZuYNCXe9XRJyk=
+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/C05ACKg1i1fjUJ5Ep61QUkGW8qpdssI0+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:mXi4GBBBbnImb6dmsKGUJ2LatrhH/nqhxcFungHvyanc=
+golang.org/x/net v0.0.0-20190311183353-d8887717615a/go.mod h1:t9HGtf8HONx5eT2rtn7q6eTqICyQUVnKs3thJo3Qp1g=
+golang.org/x/net v0.0.0-20190404232315-eb5bcb51f2a3/go.mod h1:t9HGtf8HONx5eT2rtn7q6eTqICyQUVnKs3thJo3Qp1g=
golang.org/x/net v0.0.0-20190620200207-3b0461eec859 h1:R/3boaszxf16EUWTVDzSKVwLmS3pwZ1yqXm8j0v2QI=
golang.org/x/net v0.0.0-20190620200207-3b0461eec859/go.mod h1:z5CRVTTtmAJ677TzLLGU+0bjP00Lku0Li4/5GtJWs/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/jJTE5uejppjVl0e/izrB70Jof72aM=
(patch-import-return-locations) tools/gopls $
```

```
31 "golang.org/x/tools/go/ast/astutil"
30 "golang.org/x/tools/internal/lsp/protocol"
29 "golang.org/x/tools/internal/span"
28 "golang.org/x/tools/internal/telemetry/trace"
27 errors "golang.org/x/xerrors"
26 )
25
24 // IdentifierInfo holds information about an identifier in Go source.
23 type IdentifierInfo struct {
22     Name string
21     File ParseGoHandle
20     mappedRange
19
18     Type struct {
17         mappedRange
16         Object types.Object
15     }
14
13     Declaration Declaration
12
11     pkg          Package
10     ident        *ast.Ident
9     wasEmbeddedField bool
8     qf           types.Qualifier
```

```
24 f, err := view.GetFile(ctx, uri)
23 if err != nil {
22     return nil, err
21 }
20 ident, err := source.Identifier(c
19 if err != nil {
18     return nil, err
17 }
16 decRange, err := ident.Declaratio
15 if err != nil {
14     return nil, err
13 }
12 return []protocol.Location{
11     {
10         URI: protocol.NewURI(id
9         Range: decRange,
8     },
7 }, nil
6 }
5
4 func (s *Server) typeDefinition(ctx c
TypeDefinitionParams) ([]protocol.Loc
3 uri := span.NewURI(params.TextDoc
2 view := s.session.ViewOf(uri)
```


bottomline

:set mouse=a

even if you are a power user
or keyboarder

pair programming

easy on the person next to you

unexpected surprises

no more of them

visual cues

“no thanks, i’d rather search my way through”

directory tree

netrw (:Explore)

NerdTree

```
</programming/golang/tools/
├─ benchmark/
├─ blog/
├─ cmd/
├─ container/
├─ cover/
├─ go/
├─ godoc/
├─ gopls/
├─ imports/
├─ internal/
│   ├─ apidiff/
│   ├─ fastwalk/
│   ├─ gopathwalk/
│   ├─ imports/
│   ├─ jsonrpc2/
│   ├─ lsp/
│   ├─ memoize/
│   └─ module/
│       └─ module.go
│           └─ module_test.go
├─ semver/
├─ span/
├─ telemetry/
├─ testenv/
├─ tool/
├─ txtar/
├─ xcontext/
├─ playground/
├─ present/
├─ refactor/
├─ AUTHORS
├─ codereview.cfg
├─ CONTRIBUTING.md
├─ CONTRIBUTORS
├─ go.mod
├─ go.sum
├─ LICENSE
├─ PATENTS
├─ README.md
├─ tags
├─ tags.lock
└─ tags.temp
```

NERD_tree_1

20:9

```
25     return "", err
24 }
23
22     return encodeString(path)
21 }
20
19 // EncodeVersion returns the safe encoding of the given module version.
18 // Versions are allowed to be in non-semver form but must be valid file names
17 // and not contain exclamation marks.
16 func EncodeVersion(v string) (encoding string, err error) {
15     if err := checkElem(v, true); err != nil || strings.Contains(v, "!") {
14         return "", fmt.Errorf("disallowed version string %q", v)
13     }
12     return encodeString(v)
11 }
10
9 func encodeString(s string) (encoding string, err error) {
8     haveUpper := false
7     for _, r := range s {
6         if r == '!' || r >= utf8.RuneSelf {
5             // This should be disallowed by CheckPath, but diagnose anyway.
4             // The correctness of the encoding loop below depends on it.
3             return "", fmt.Errorf("internal error: inconsistency in EncodePath")
2         }
1         if 'A' <= r && r <= 'Z' {
464             haveUpper = true
1         }
0         }
3
4     if !haveUpper {
5         return s, nil
6     }
7
8     var buf []byte
9     for _, r := range s {
10         if 'A' <= r && r <= 'Z' {
11             buf = append(buf, '!', byte(r+'a'-'A'))
12         } else {
13             buf = append(buf, byte(r))
14         }
15     }
16     return string(buf), nil
17 }
18
```

NORMAL

patch-import-return-locations | module.go

sign column

linter/formatter

git integration

```
10 func encodeString(s string) (encoding string, err error) {
-   9   for _, r := range s {
-     8       if r == '!' || r >= utf8.RuneSelf {
-       6           // This should be disallowed by CheckPath, but diagnose anyway.
-       5           // The correctness of the encoding loop below depends on it.
-       4           return "", fmt.Errorf("internal error: inconsistency in EncodePath")
-       3       }
-     2   }
-     1
- 464   if 'A' <= r && r <= 'Z' {
+     1       haveUpper = true
+     2   }
+     3
+     4   if !haveUpper {
+     5       return s, nil
+     6   }
+     7
+     8   var buf []byte
+     9   for _, r := range s {
+    10       if 'A' <= r && r <= 'Z' {
+    11           buf = append(buf, '!', byte(r+'a'-'A'))
+    12       } else {
>> 13           buf = append(buf, byte(r))
>> 14       }
+    15   }
>> 16   return string(buf), nil
>> 17 }
18
19 // DecodePath returns the module path of the given safe encoding.
20 // It fails if the encoding is invalid or encodes an invalid path.
21 func DecodePath(encoding string) (path string, err error) {
22     path, ok := decodeString(encoding)
>> 23     if !ok {
>> 24         return "", fmt.Errorf("invalid module path encoding %q", encoding)
>> 25     }
>> 26     if err := CheckPath(path); err != nil {
>> 27         return "", fmt.Errorf("invalid module path encoding %q: %v", encoding, err)
>> 28     }
>> 29     return path, nil
>> 30 }
31
32 // DecodeVersion returns the version string for the given safe encoding.
33 // It fails if the encoding is invalid or encodes an invalid version.
NORMAL patch-import-return-locations | module.go | +
```

status line

branch info

filename

cursor row/column

language encoding

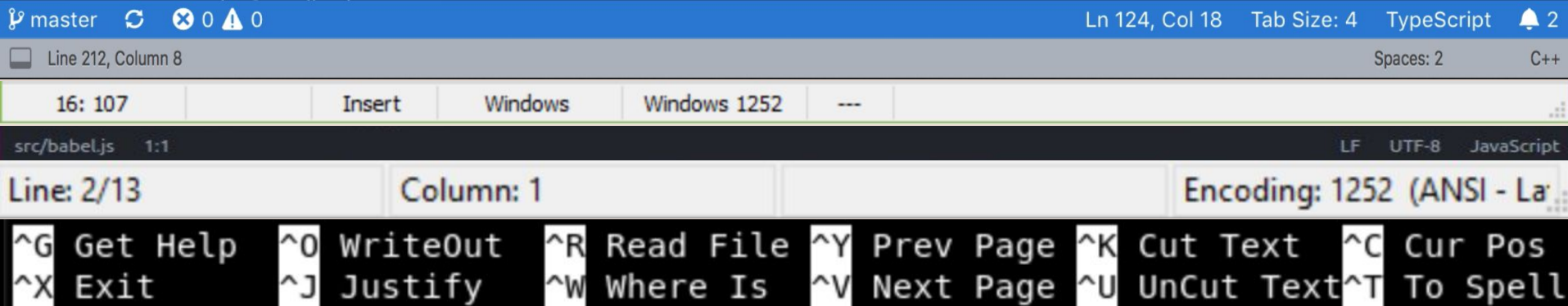
tabs/spaces

editing mode

linting errors

buffer identifier

filetype



status line

custom

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    }
```

additional

“but i’m good with fuzzy search”

code navigation

tags

ctags, exuberant ctags,
universal ctags

language server protocol

ale, vim-lsp, coc.nvim

search

fzf, ctrl-p

thank you
ありがとう

@danishprakash

references

- a. reference 1
- b. reference 2