

designing concepts: purposes, OPs, actions & states

Daniel Jackson · Autodesk · Woodinville, WA · Dec 3-5, 2024

purposes
& OPs

seeking a UI-independent definition

This is homework and I'm having a
are the definitions of the objects:

8

```
sig Library {  
    patrons : set Person,  
    on_shelves : set Book,  
}
```

1

StackOverflow

Reader Picks All

J John Boston | Oct. 27

To protect children? Seems far more likely it's yet one more way to extract personal information to feed the insatiable advertising machines

1 Reply 143 Recommend Share Flag

NYTimes

Oliver Jumpertz @oliverjumpertz · Aug 30

Don't only learn programming languages and frameworks.

Also, learn:

- Problem-solving
- Critical thinking
- People skills
- **Software design** and architecture
- Computer science fundamentals

Languages and frameworks are a good start, but the above will make you even better.

49 392 1,935

Twitter

#1: give it a name

concept Upvote

what other names might you choose?

why do names matter?

#2: say what it's for (purpose)

concept Upvote

purpose rank items by popularity

purpose encourage authors

purpose engage evaluators

why is it important to know your purpose?

why's it good to identify a primary purpose?

what is the design impact of one purpose over another?

similar UIs, very different purposes

concept Upvote

purpose rank items by popularity

▲ This is homework and I'm having a
are the definitions of the objects:
8
▼
1

```
sig Library {  
    patrons : set Person,  
    on_shelves : set Book,  
}
```

concept Reaction

purpose support quick responses

Today ▾
Daniel I think we should organize a
software concepts forum.
 

concept Recommendation

purpose infer user preferences



#3: explain how it works (operational principle)

concept Upvote

purpose rank items by popularity

principle after series of upvotes
of items, the items are ranked by
their number of upvotes

operational principles for related concepts

concept Upvote

purpose rank items by popularity

principle after series of upvotes of items, the items are ranked by their number of upvotes



This is homework and I'm having a
are the definitions of the objects:

8



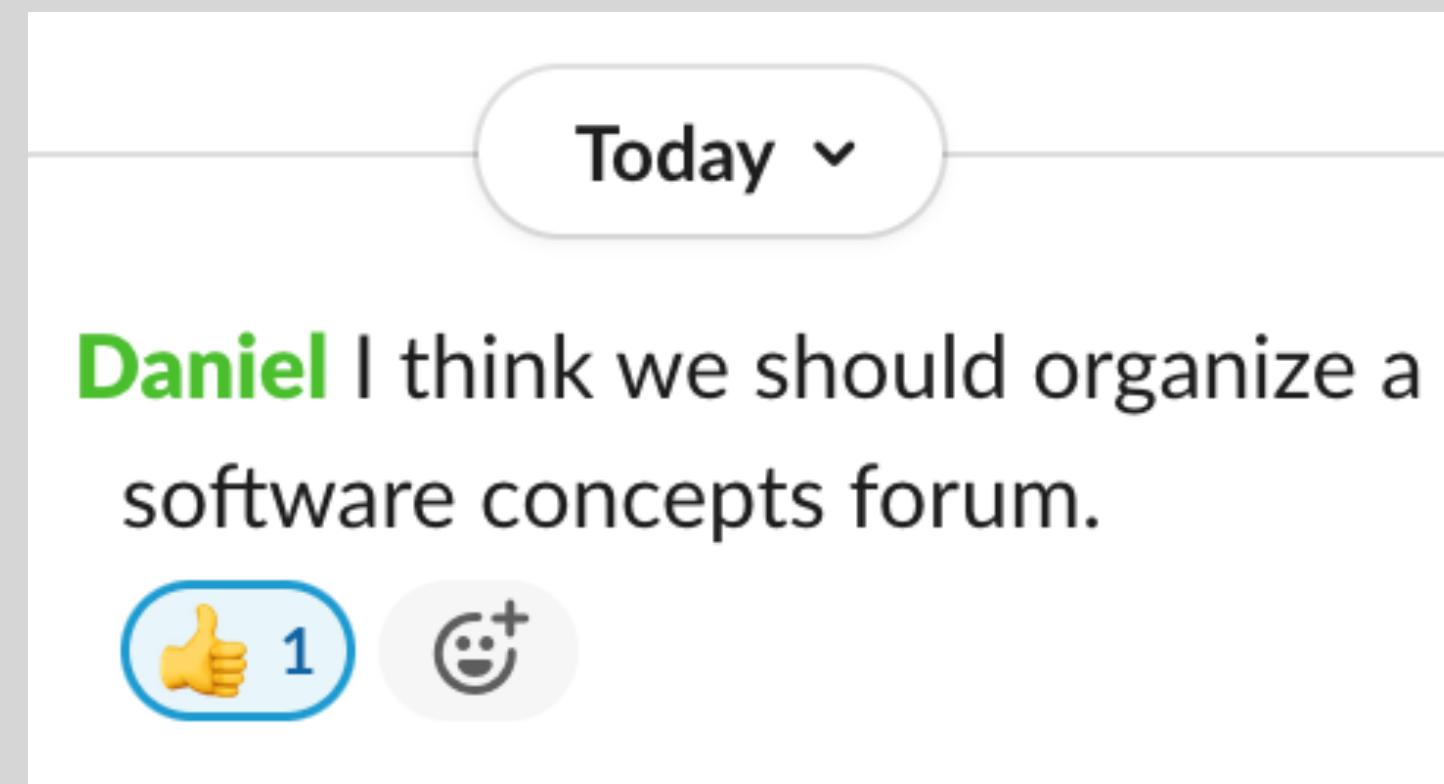
```
sig Library {  
    patrons : set Person,  
    on_shelves : set Book,  
}
```

1

concept Reaction

purpose support quick responses

principle when user selects reaction, it's shown to the author (often in aggregated form)



concept Recommendation

purpose infer user preferences

principle user's likes lead to ranking of kinds of items, determining which items are recommended



check your
understanding

Event Creation Home

Welcome, DNJ

You have
6 future meetings
spanning 53 days.

November 2024						
Su	Mo	Tu	We	Th	Fr	Sa
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

Find a Meeting:

ANSWER

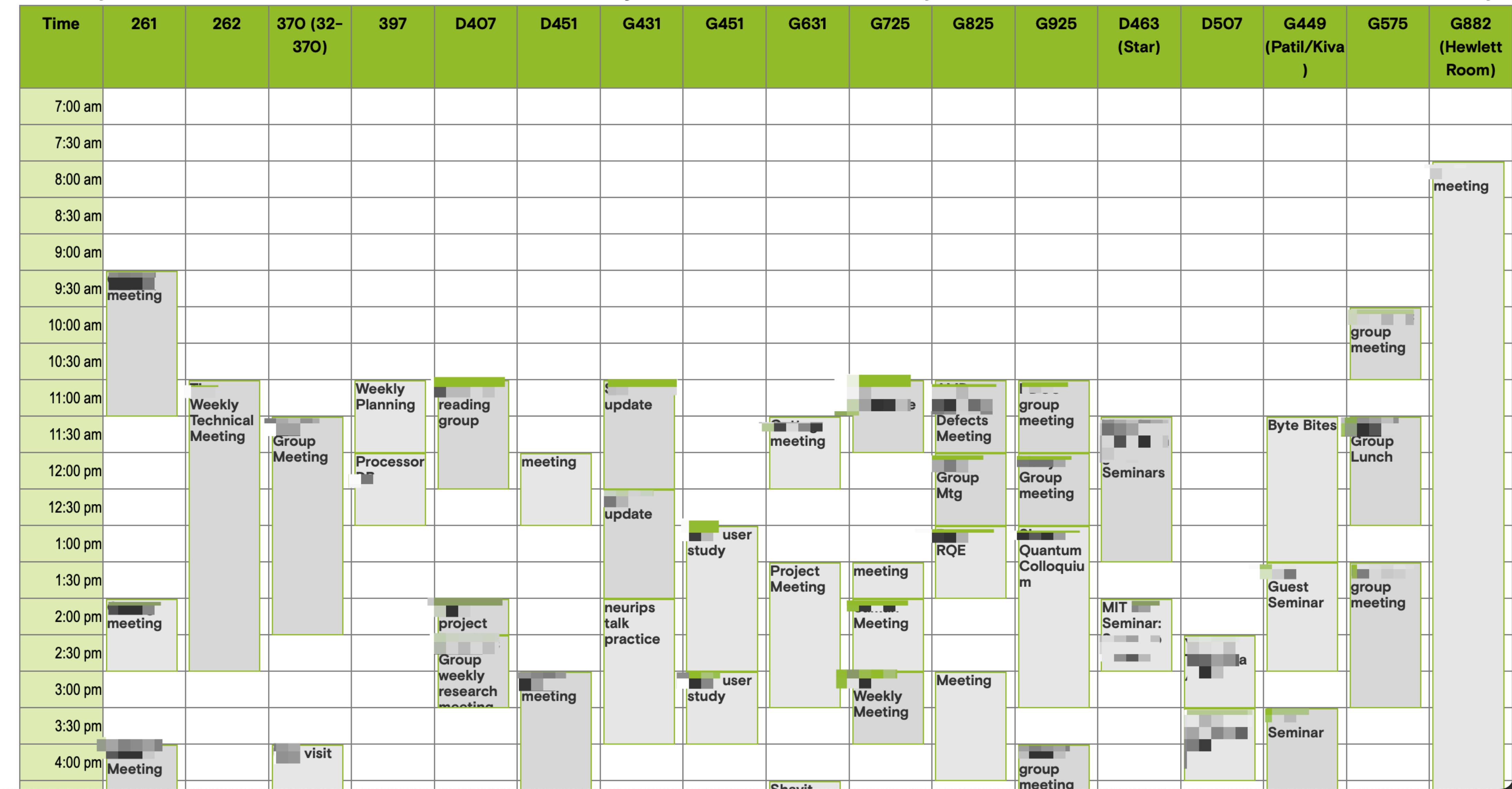
Go

CSAIL Meetings

« Monda

Tuesday 12 November 2024 (Go to Today)

Wednesday »



check your understanding: purposes

consider the concept

ConferenceRoomBooking

which of these is a good purpose?

- “make it easy to book conference rooms”
- “address Autodesk’s conference room demand”
- “ensure equitable use of conference rooms”
- “prevent conflicts in conference room usage”
- “improve conditions for in-person work”
- “manage orderly allocation of conference rooms”
- “ease process of finding conference room for meeting”
- “ensure conference room availability when needed”
- “help employees get rooms and prevent others”

how generic or specific is this concept?

conference rooms or any room?

rooms or any resource?

checklist: purposes

- cogent, not vague
- need-focused
- specific
- evaluable & refutable
- one purpose, not more

check your understanding: OPs

which of these is a good OP?

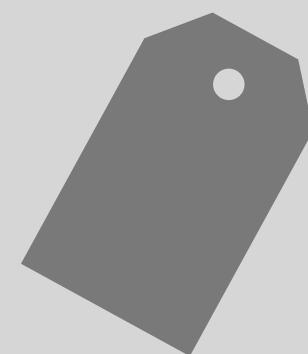
- “when you select a slot, enter info and click submit, the booking appears”
- “when you book a room, it appears in the calendar”
- “if you book a room for a time slot, it will be available for you to use then”
- “to use a conference room, first you have to book a slot”
- “when you book a slot, nobody else can take it afterwards”

checklist: OPs

- ✓ UI independent
- ✓ value: end-to-end
- ✓ compelling story
- ✓ matches purpose
- ✓ minimal context

defining behavior
with states & actions

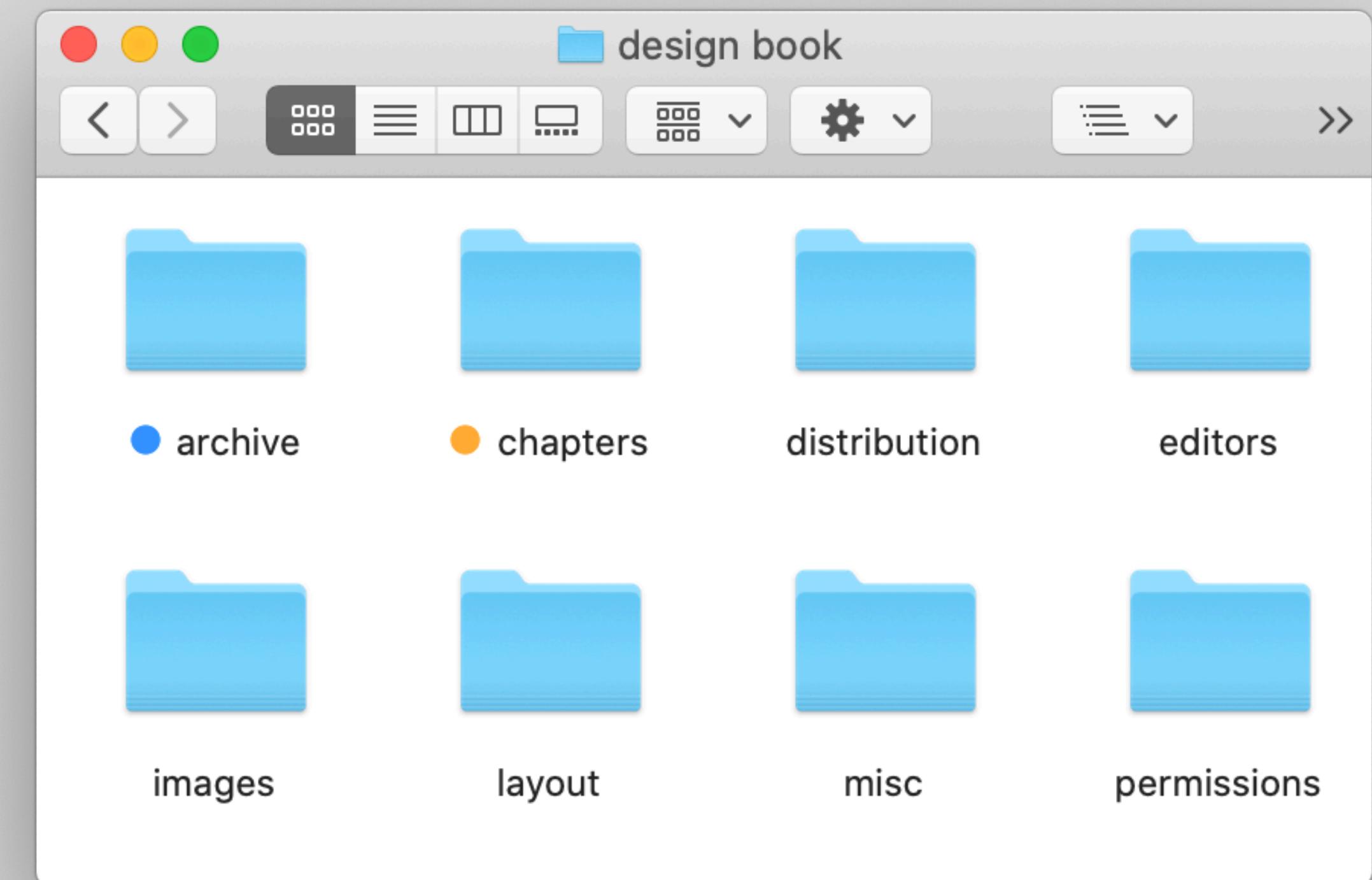
a simple but potent concept



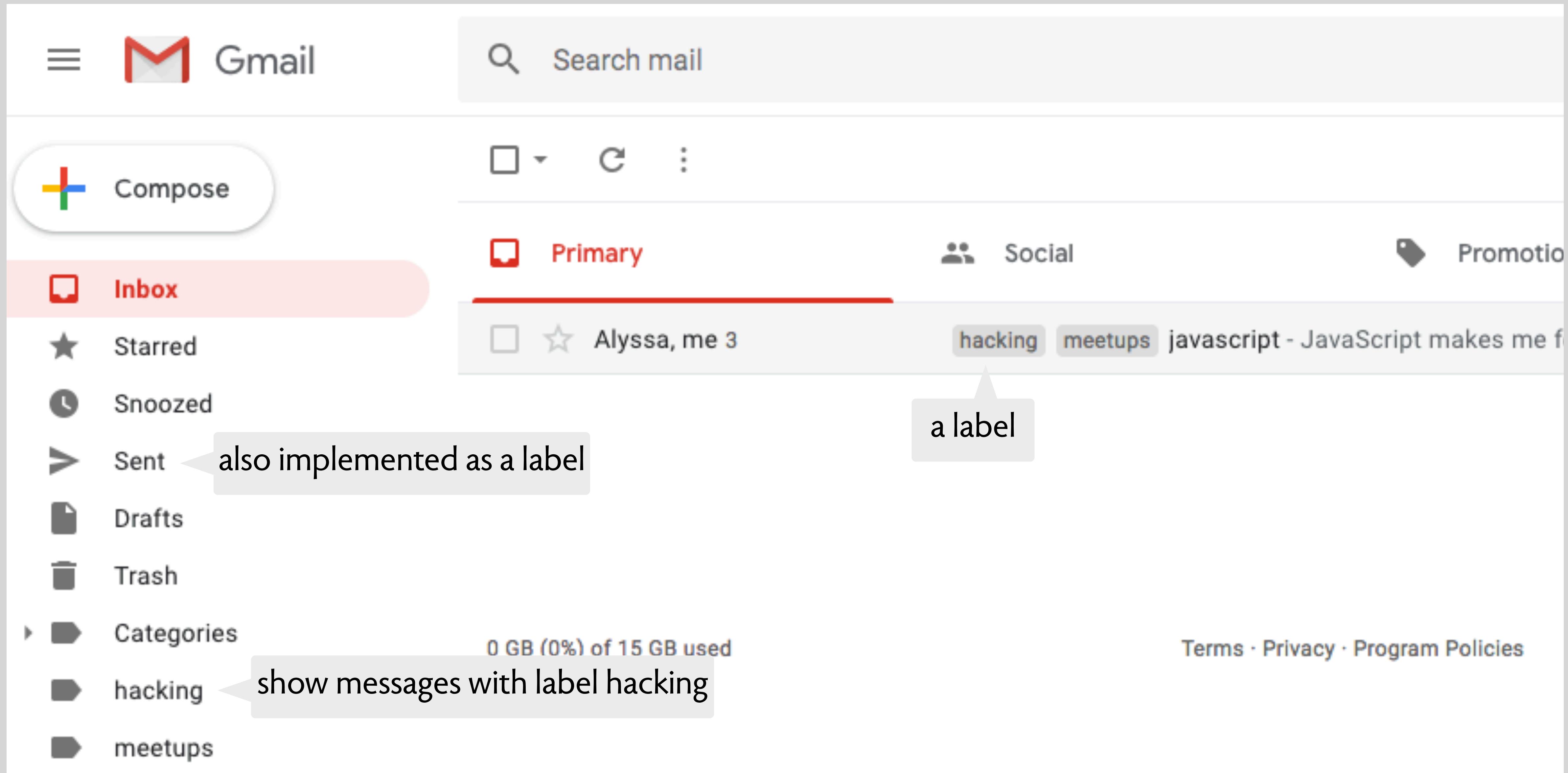
concept Labeling

purpose organize items

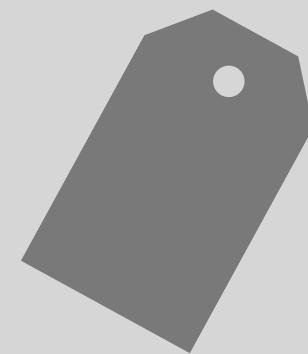
principle if you add a label to an item, then later you can filter on that label and find the item



another application: the labeling concept in Gmail



defining the concept's actions



concept Labeling

purpose organize items

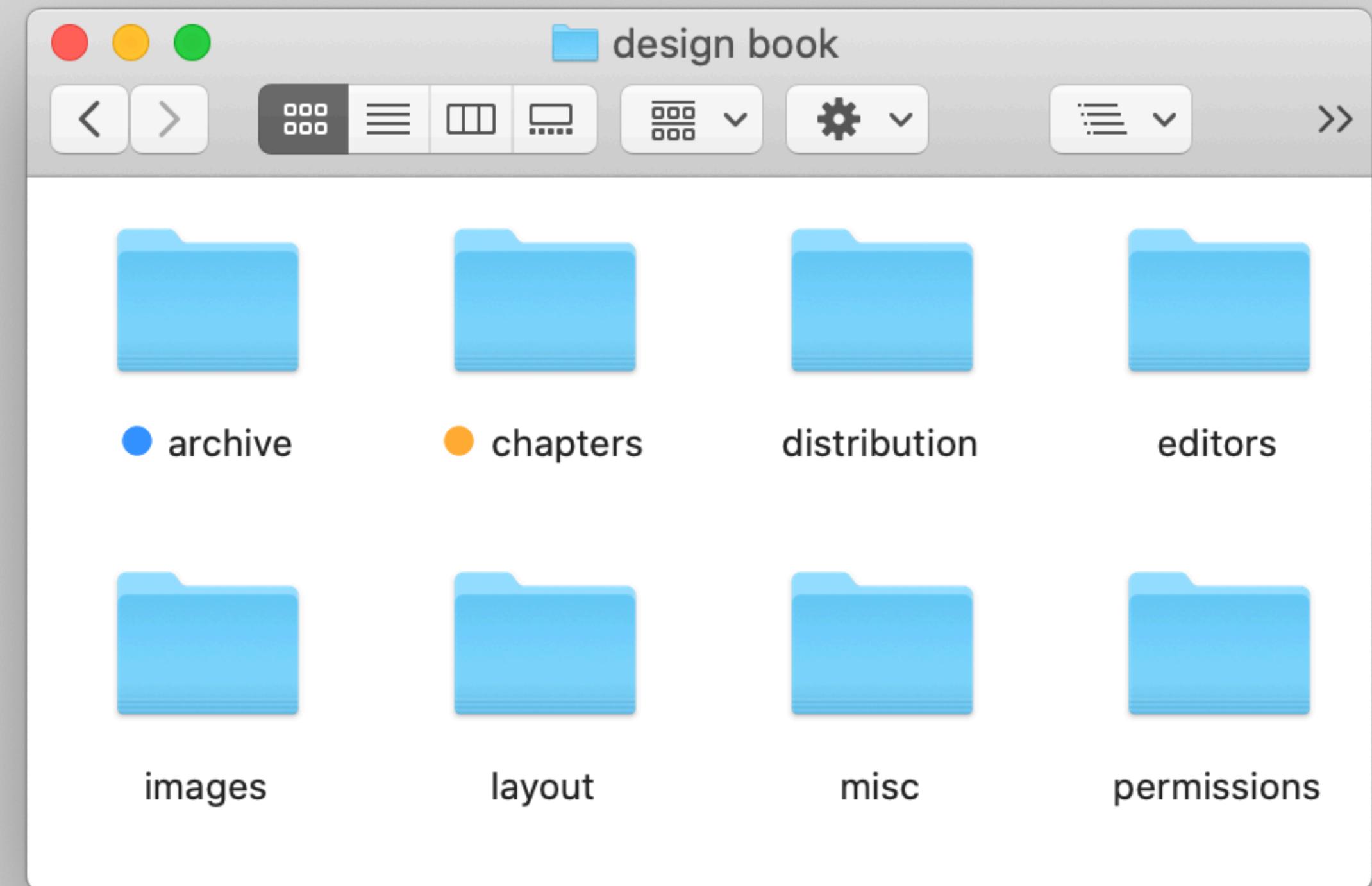
principle if you add a label to an item, then later you can filter on that label and find the item

actions

add label to an item

remove label from an item

filter on a set of labels



defining the concept's state



concept Labeling

purpose organize items

principle if you add a label to an item, then later you can filter on that label and find the item

state

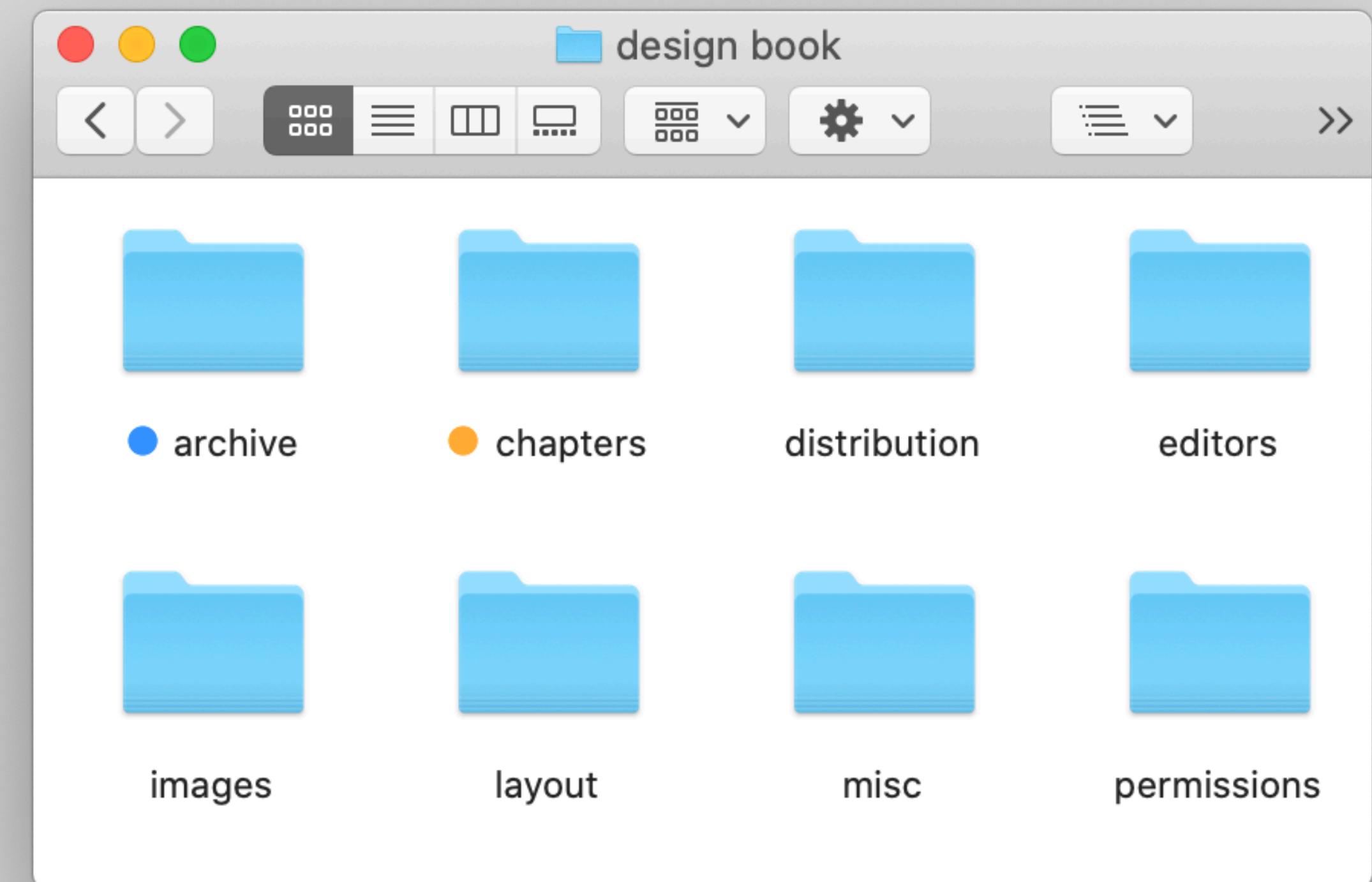
labels for each item

actions

add label to an item

remove label from an item

filter on a set of labels



making the state precise



a type variable

concept Labeling [Item]

concept is generic

purpose organize items

principle if you add a label to an item, then later you can filter on that label and find the item

state

labels: Item -> **set Label**

actions

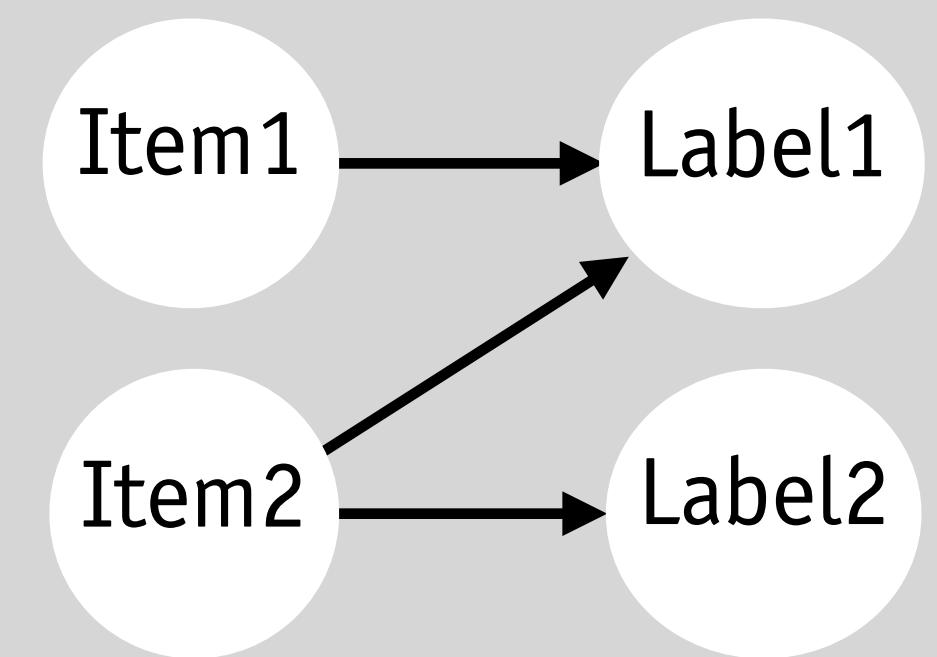
add label to an item

remove label from an item

filter on a set of labels



Item1	Label1
Item2	Label1
Item2	Label2



making the actions precise



concept Labeling [Item]

purpose organize items

principle if you add a label to an item, then later you can filter on that label and find the item

state

labels: Item \rightarrow set Label

actions

add (l: Label, i: Item)

i.labels += l *add l to the labels of i*

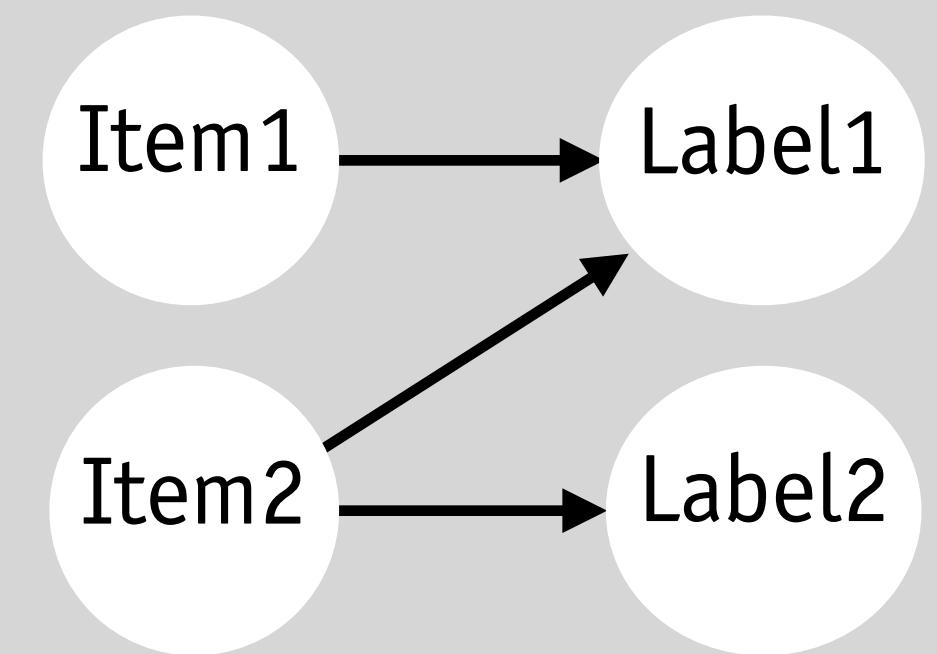
remove (l: Label, i: Item)

i.labels -= l

filter (ls: set Label): set Item

return {i: Item | ls in i.labels}

Item1	Label1
Item2	Label1
Item2	Label2



check your understanding: how does an action update the state?



concept Labeling [Item]

purpose organize items

principle if you add a label to an item, then later you can filter on that label and find the item

state

labels: Item -> **set** Label

actions

add (l: Label, i: Item)

i.labels += l

remove (l: Label, i: Item)

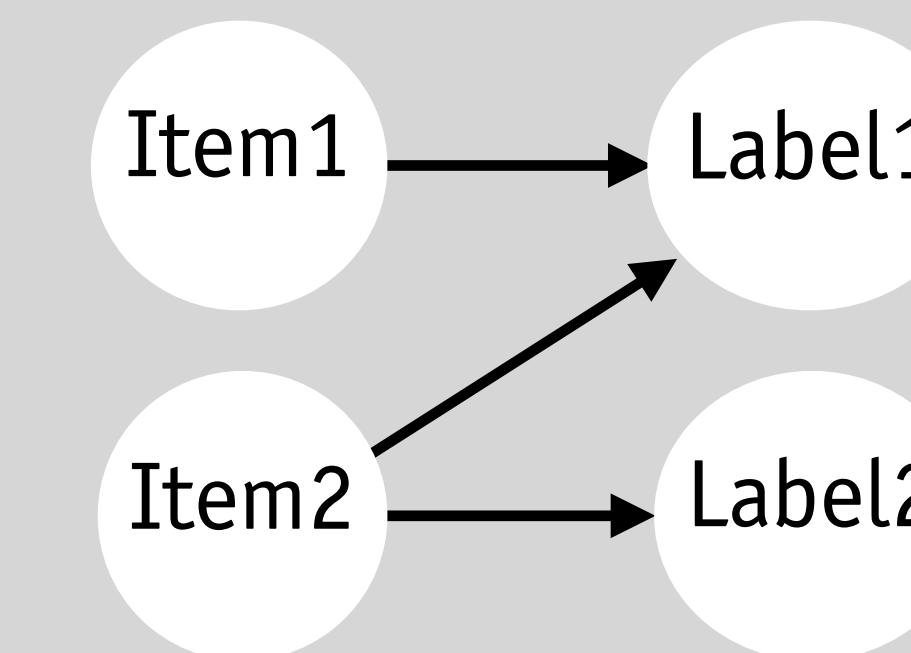
i.labels -= l

filter (ls: **set** Label): **set** Item

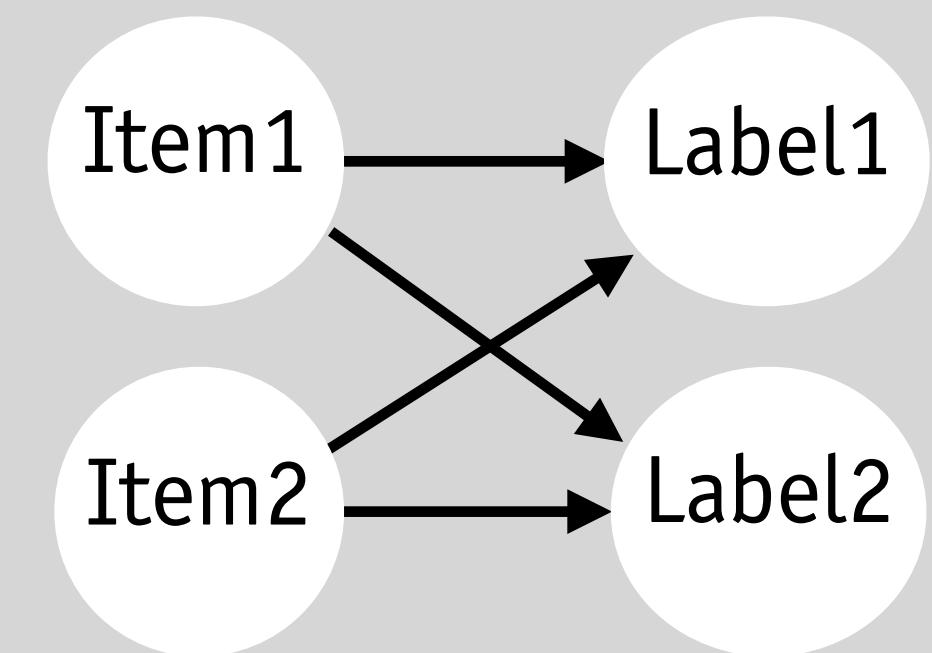
return {i: Item | ls **in** i.labels}

Item1	Label1
Item2	Label1
Item2	Label2

Item1	Label1
Item1	Label2
Item2	Label1



before add (Label2, Item1)



after add (Label2, Item1)

checklists:
states & actions

check your understanding: what's a good action?



concept Labeling [Item]

purpose organize items

principle if you add a label to an item, then later you can filter on that label and find the item

state

labels: Item -> **set** Label

actions

add (l: Label, i: Item)

i.labels += l

remove (l: Label, i: Item)

i.labels -= l

filter (ls: **set** Label): **set** Item

return {i: Item | ls **in** i.labels}

checklist: actions

- ✓ high level, not UI
- ✓ atomic, not ongoing
- ✓ user facing, not internal
- ✓ cover the OP and more

which of these might be a reasonable action?

select item to label

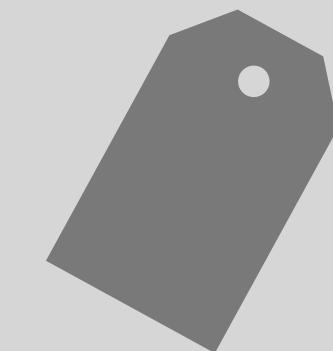
clear storage for unused labels

remove all labels from item

copy labels from one item to another

maintain display of item labels

check your understanding: what's a good state?



concept Labeling [Item]

purpose organize items

principle if you add a label to an item, then later you can filter on that label and find the item

state

labels: Item -> **set** Label

actions

add (l: Label, i: Item)

i.labels += l

remove (l: Label, i: Item)

i.labels -= l

filter (ls: **set** Label): **set** Item

return {i: Item | ls **in** i.labels}

checklist: states



sufficient for actions



no implementation bias



no useless information

which of these might be a reasonable state?

a hash table mapping items to lists of labels

a set of items and a set of labels

a set of labelings, each being an item and a label

a mapping from labels to sets of items

check your understanding: which state is correct?

concept User

state

username: UserName

password: Password

concept UserAuth

state

username: User -> **one** UserName

password: User -> **one** Password

checklist: states



sufficient for actions



no implementation bias



no useless information

check your understanding: what actions & states for Upvote?

concept Upvote [Item, User]

make generic

purpose rank items by popularity

principle after series of upvotes of items, the items are ranked by their number of upvotes

state

by: Vote -> **one** User

for: Vote -> **one** Item

Upvote, Downvote: **set** Vote

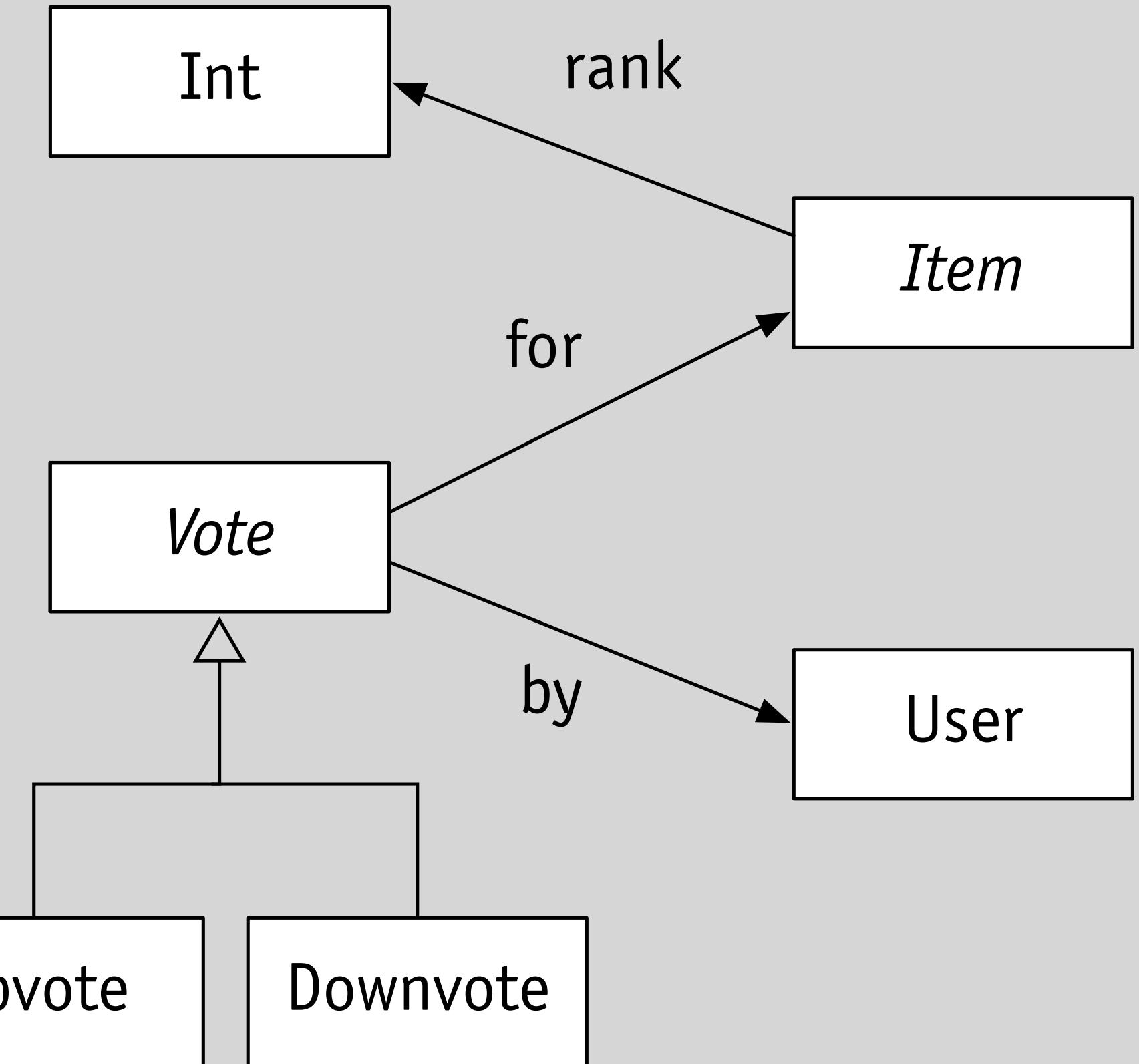
rank: Item -> **one** Int

actions

upvote (u: User, i: Item)

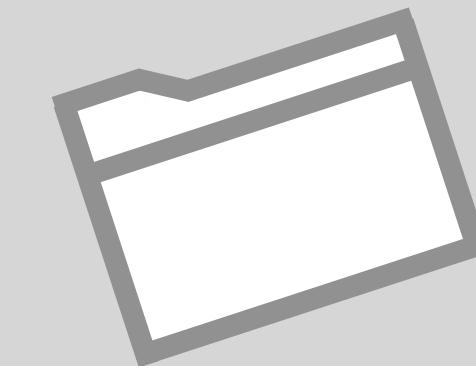
downvote (u: User, i: Item)

unvote (u: User, i: Item)



designing
richer states

a simple folder concept, as used in imap



concept FolderTree

state

a root folder

for each folder

the folders or items it contains
a name for each folder

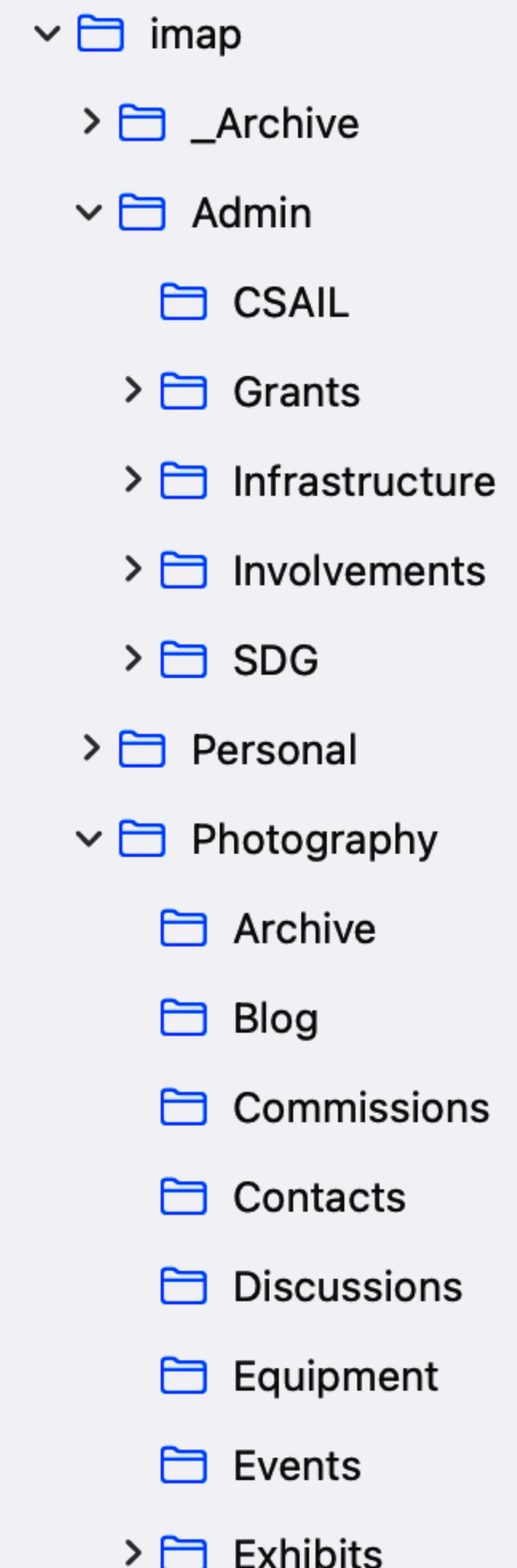
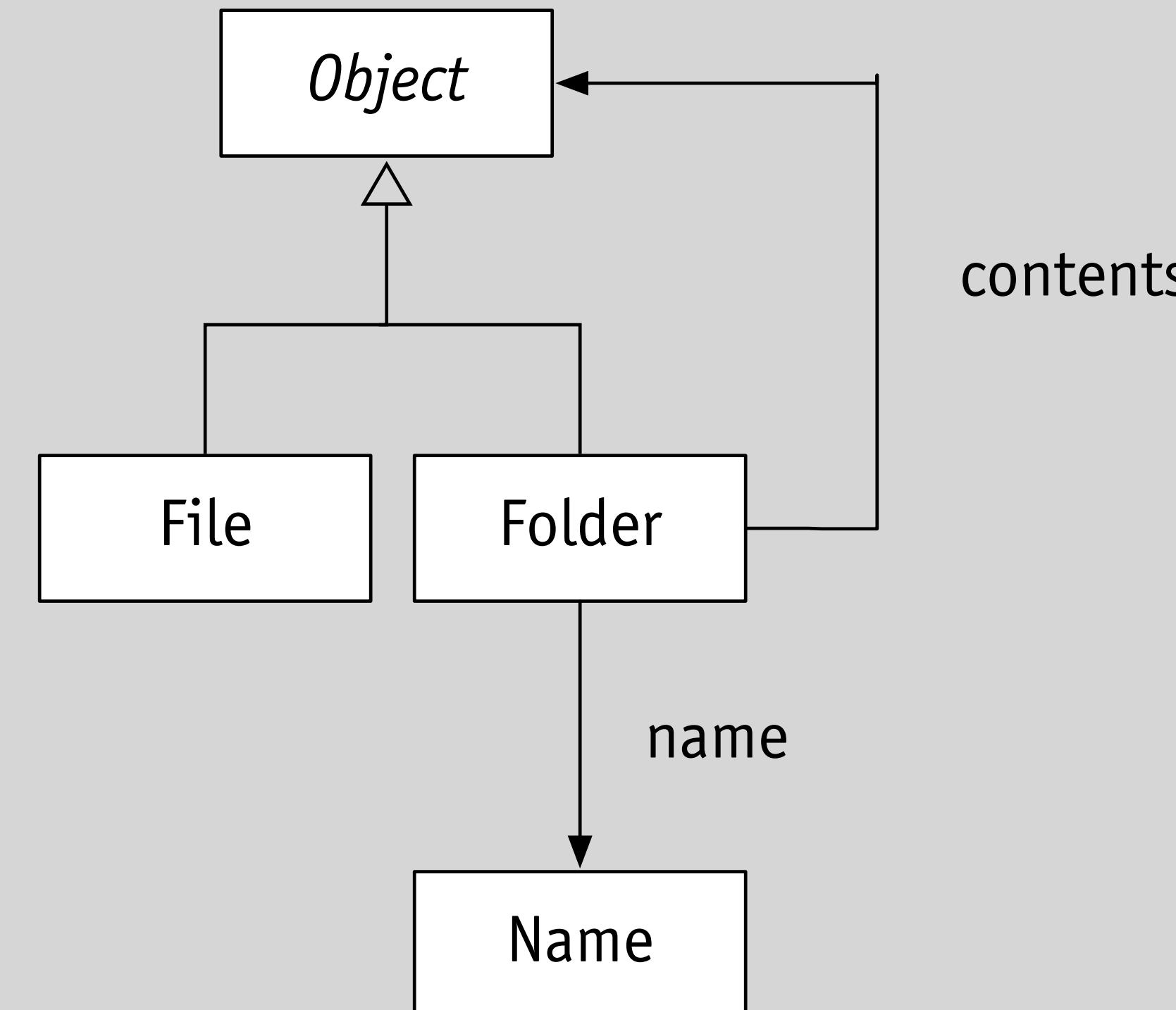
actions

create a folder

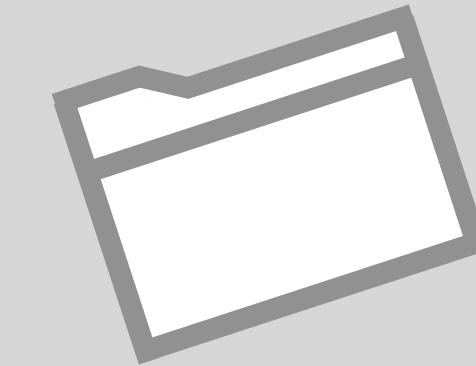
delete a folder

move an object to a folder

rename a folder



exercise: which states are valid?



concept FolderTree

state

a root folder

for each folder

the folders or items it contains
a name for each folder

what are the invariants (rules) of this state?

no folder or item has two parents

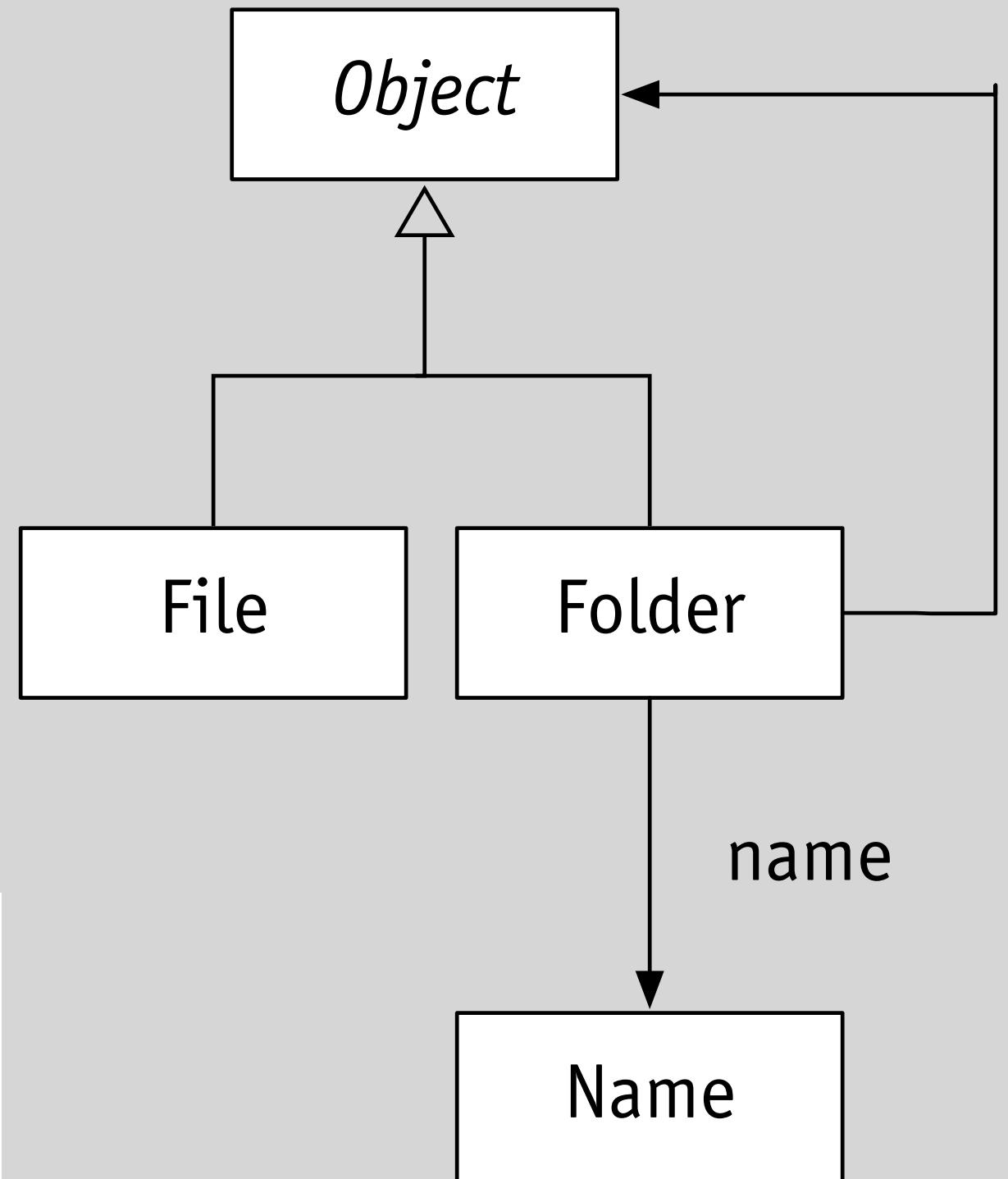
every folder or item (except the root) has a parent

no two folders have the same name

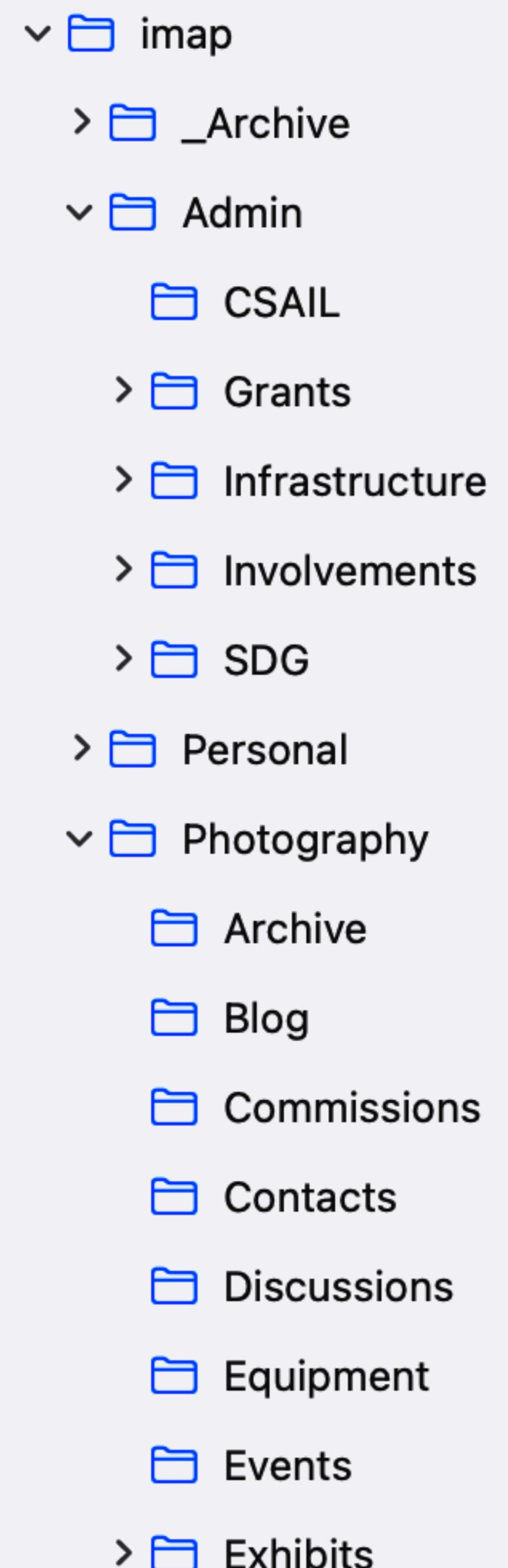
a folder can contain either items or folders, not both

no folder can contain itself, directly or indirectly

all objects reachable from the root



contents



exercise:
an Autodesk concept

defining a concept

in pairs, pick a concept from an Autodesk product
give it a name

write an OP

produce a list of actions, starting from the OP

devise a state to support the actions

outcome-based BIM concepts

listed on next slide

Outcome-based BIM Concepts

Created by Jason Oliveira, last modified by Angie Peng on Sep 06, 2024

- **Benchmark** — *The purpose of a Benchmark in Analysis workflows* is to store a named value for comparison with the value of a Metric.
- **Factor** — *The purpose of a Factor in the context of Analysis workflows* is to identify and store the value of a variable upon which one or more Metrics depend.
- **Outcome** — *The purpose of an Outcome is* to describe the ultimate result of a Project.
- **Proposal - Forma Design Client** — The purpose of a proposal is to track multiple options for a site.

They are intended to represent different choices for modelling the entire site, not just optioneering within the smaller scale, like what facade to have on a building or how to organise a section of the site.

A proposal presents as an element in the *element system*, and follows those semantics for modelling project data.

- **Scenario** — *The purpose of Scenarios in Analysis workflows* is to associate Analysis inputs with corresponding Analysis results for a single Analysis type and run.
- **Target** — *The purpose of a Target in Analysis workflows* is to store the desired value or value range for a Metric, often relative to one or more Benchmarks.

checklist: purposes

- ✓ cogent, not vague
- ✓ need-focused
- ✓ specific
- ✓ evaluable & refutable
- ✓ one purpose, not more

checklist: OPs

- ✓ UI independent
- ✓ value: end-to-end
- ✓ compelling story
- ✓ matches purpose
- ✓ minimal context

checklist: actions

- ✓ high level, not UI
- ✓ atomic, not ongoing
- ✓ user facing, not internal
- ✓ cover the OP and more

checklist: states

- ✓ sufficient for actions
- ✓ no implementation bias
- ✓ no useless information

takeaways



how would you explain the concept of restaurant reservation?

to someone time-traveling from the 1850s, say

before explaining how, you might explain why

often when I come with my party, there's no room!
now restaurants have a way to ensure there will be

the
purpose



how would you explain how it works?

what you wouldn't say

press these buttons to call the restaurant, then ask for reservations...
or: the restaurant has a book listing people, times and phone numbers...

the UI

the data model

instead you might tell a story like this

if you call the restaurant and reserve a table for a certain time,
then if you turn up at that time, a table will be available for you

the OP

how would you teach a restaurant how to do it?

keep a reservation book

two columns, time slots & people

the state

instruct the staff

how to record a reservation

Reserving a Table:

When someone calls on
on ~~the~~ telephone regls and
and ~~res~~ and and requestes
available time ~~im~~
~~accione~~ it, confirm it
and wrlcm yit
wite in the book.

the actions

checklist: purposes

- ✓ cogent, not vague
- ✓ need-focused
- ✓ specific
- ✓ evaluable & refutable
- ✓ one purpose, not more

checklist: OPs

- ✓ UI independent
- ✓ value: end-to-end
- ✓ compelling story
- ✓ matches purpose
- ✓ minimal context

checklist: actions

- ✓ high level, not UI
- ✓ atomic, not ongoing
- ✓ user facing, not internal
- ✓ cover the OP and more

checklist: states

- ✓ sufficient for actions
- ✓ no implementation bias
- ✓ no useless information

what's next?

how are concepts assembled?

and how to avoid procedure calls, compromising independence?
and to allow different architectural patterns of event handling?