

Introduction

Note Title

Welcome to CSE132!

Prof. Ken Goldman (call me "Ken" or "Dr. Goldman" or ...)

Jolley Hall, Room 512

(314) 935-7542

KJG@CSE.WUSTL.EDU

Office hours: Thursdays 10-11:30 and by appointment

Course web page: <http://www.cse.wustl.edu/~kjg/cse132>

This course has a completely different character from CSE131.

Course Aims: In this course, you will develop

- Design skills
- Implementation skills
- Teamwork skills
- Communication skills
- Technical sophistication

CSE131 was a foundations course —

technical content mastered through directed exercises

CSE132 is a studio course —

- semester long project with "learning on demand"
- in teams
- lots of critique of designs and implementation

Lectures set the general direction
and readings/quizzes develop technical content

...but the real learning happens in lab.



Semester Project: An educational game for special-needs children

Some children with special needs

- have trouble communicating
- do well with assistive devices
- may have less than perfect visual acuity
- may have motor difficulties
- become frustrated easily, esp. when communication fails

Learning letters and sounds enables more computer-based communication.

Oversized keyboard, color coded, with trackball mouse.



But mainstream educational software

- ① is visually cluttered
- ② isn't easily customizable for the needs of the child, &
- ③ doesn't take advantage of assistive peripheral devices.

You will be building a game to teach letters & sounds that

- ① is visually simple
- ② is customizable/programmable by parents & teachers, &
- ③ takes advantage of the assistive keyboard.

You will build the project in stages

- • An audio clip manager
- • The basic game
- • Programming features
- • Networked interactive support

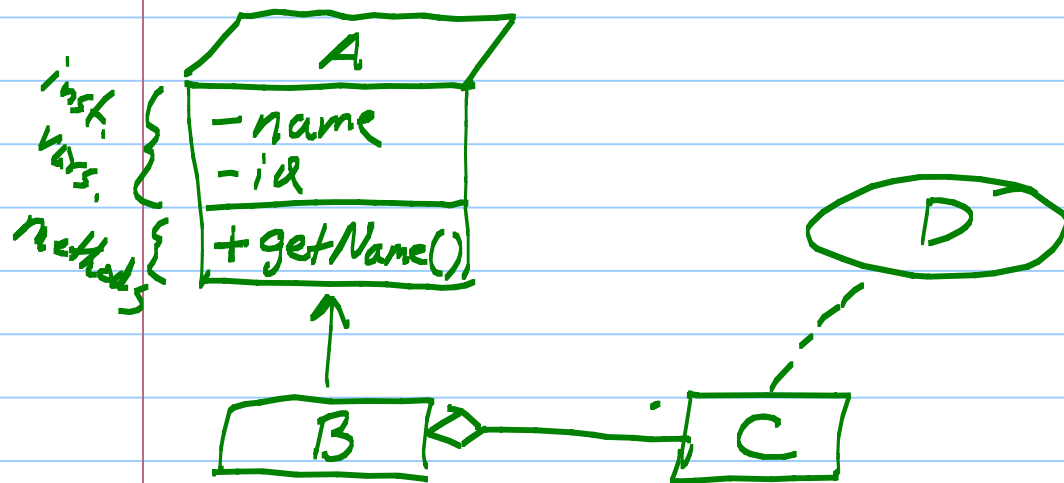
} Technical background
will be introduced
as needed to
support development.

This week, you will design the user interface for the audio clip manager.

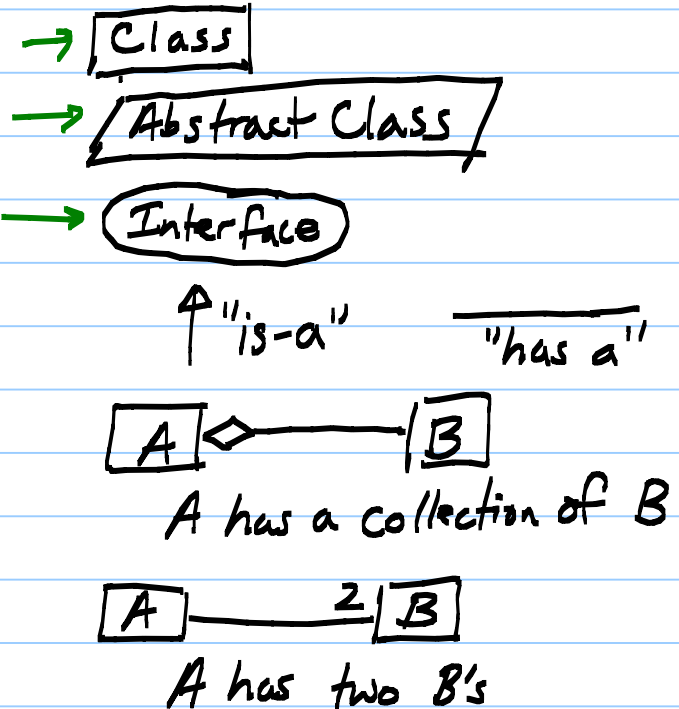
Background:

- Unified Modeling Language (UML)
- The Composite Design Pattern (example in swing)
- Common visual components of user interfaces
- Orthogonality in design — Example: Layout Managers

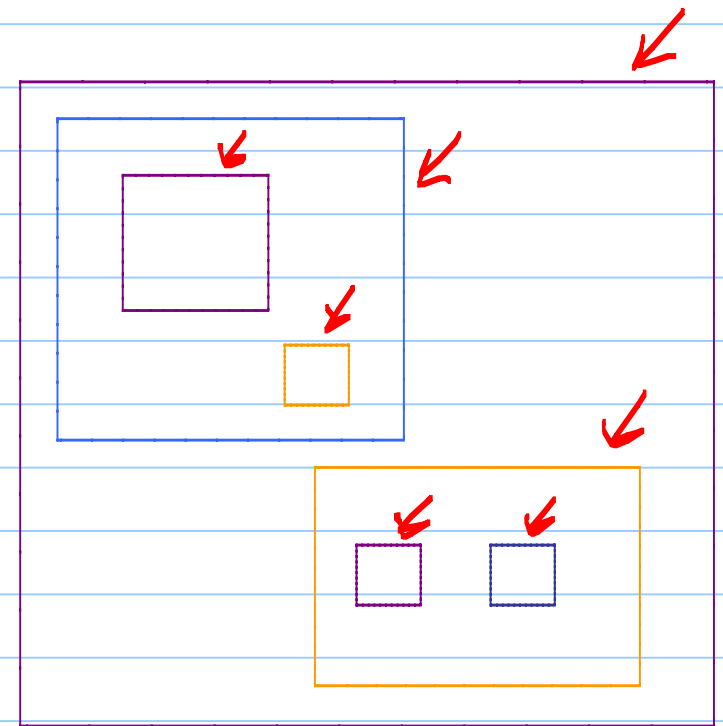
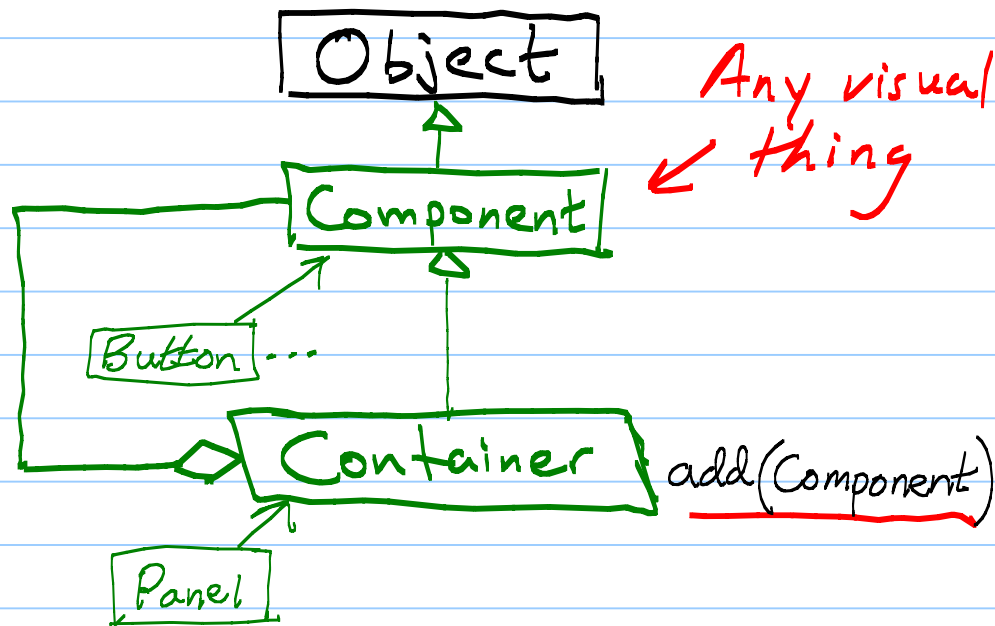
UML - Unified Modeling Language



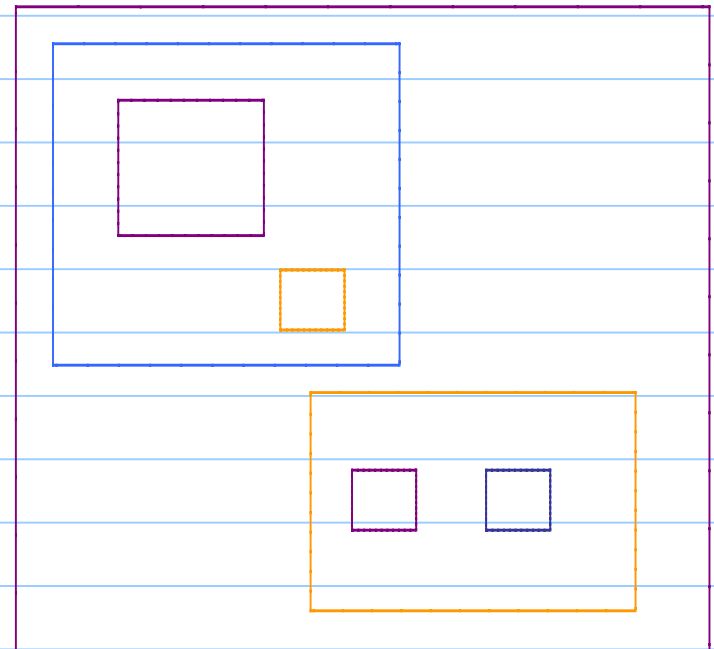
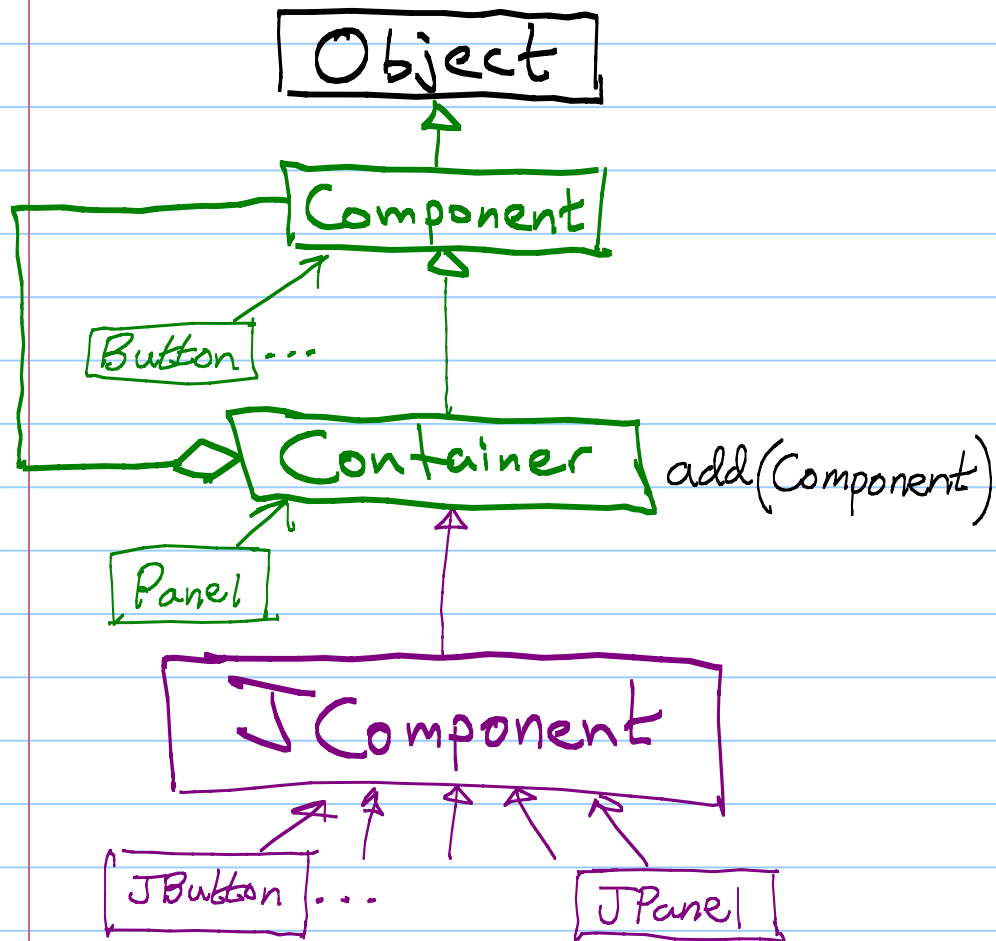
+ public (in the API)
- private
* protected



UML Case Study: The Composite Pattern



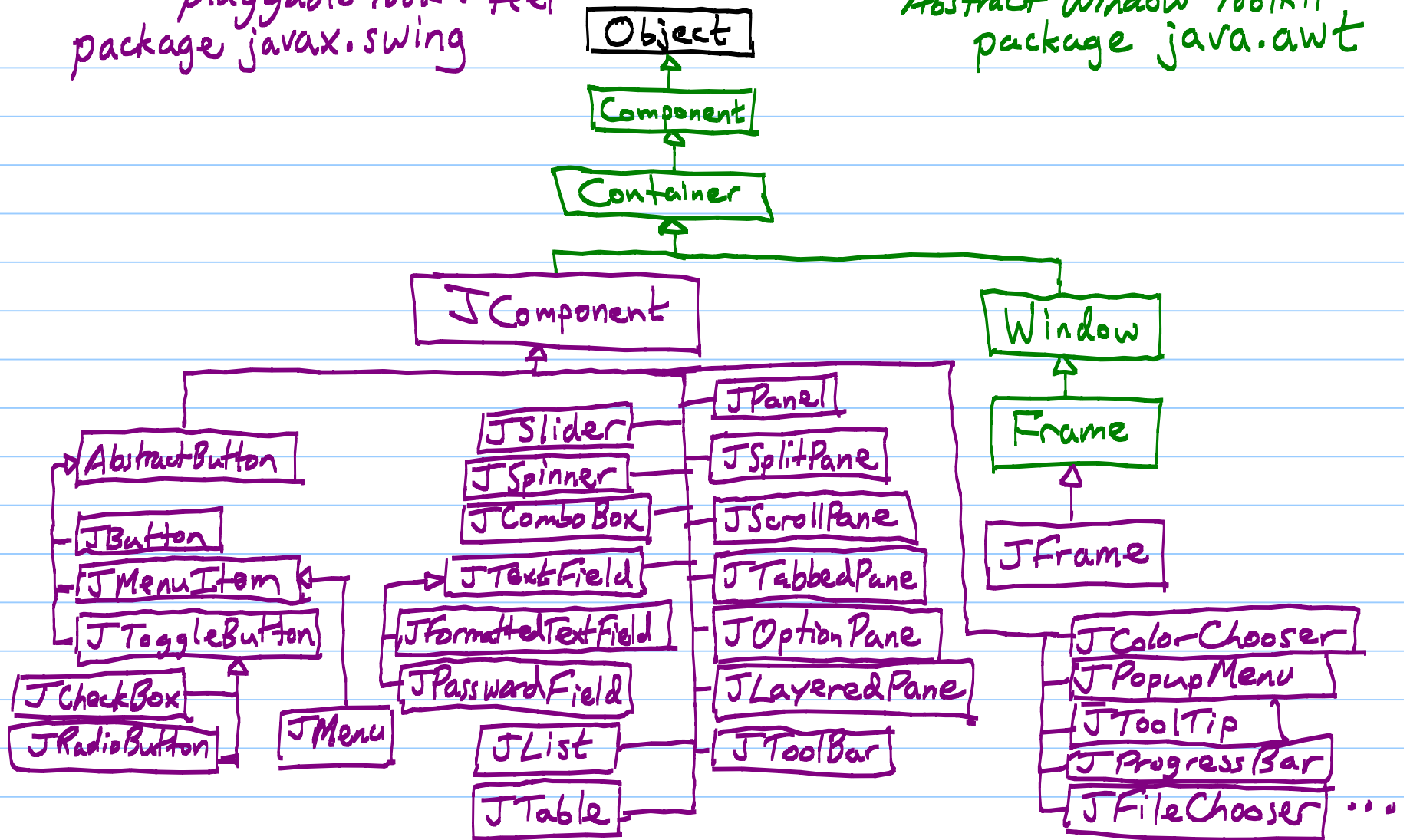
In the swing package, all components are containers!



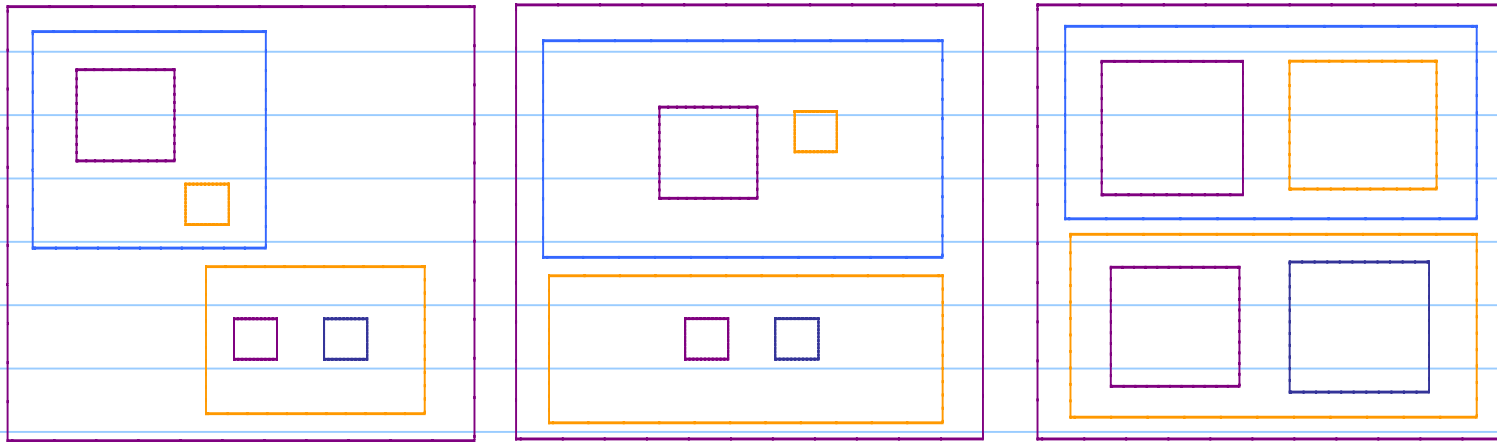
Java's view components (swing / awt)

"pluggable look & feel"
package javax.swing

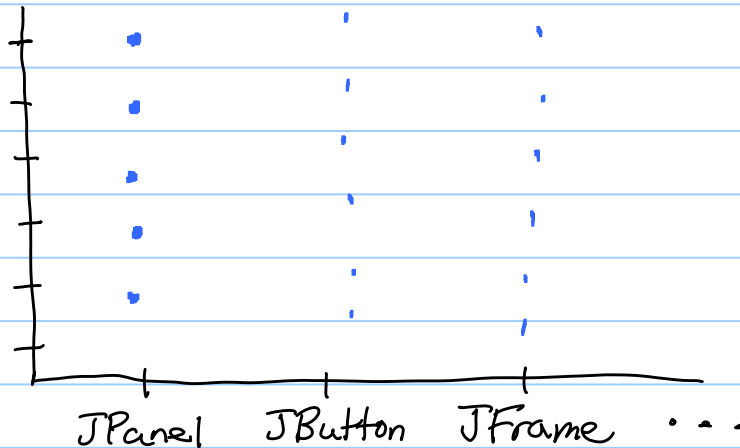
"Abstract Window Toolkit"
package java.awt



Orthogonality in design — Example: Layout



- position manually
- center in lines
- regular grid
- NSEW
- aligned
- ⋮



Solution: Layout Managers

- Container has a method

setLayout(LayoutManager mgr)

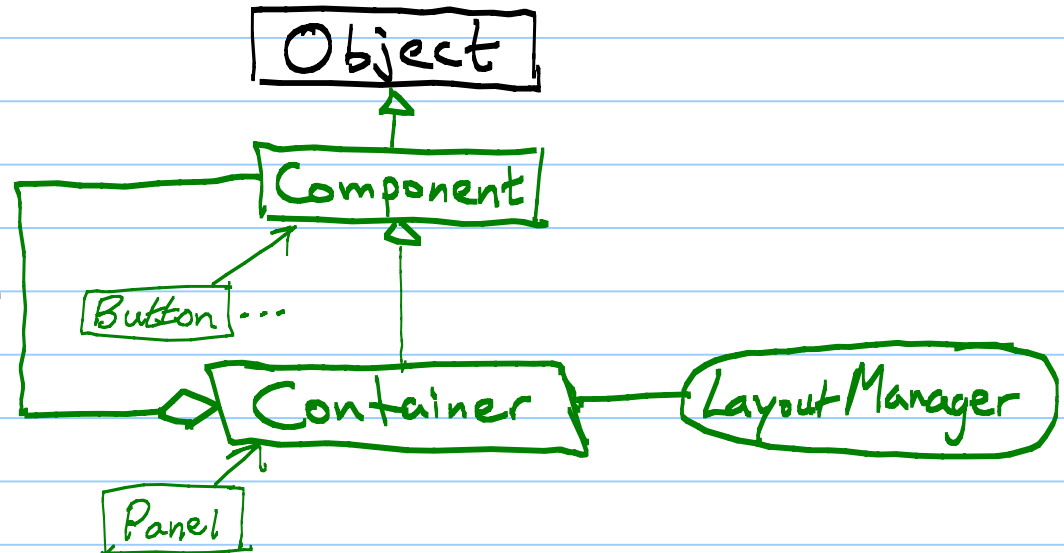
- There is a variety of provided layout managers...



NSEW

FlowLayout
GridLayout
BorderLayout
⋮

... and you can write your own.



But...

having pretty components does NOT imply that a UI is

- • easy & intuitive to learn
- easy & comfortable to use
- aesthetically pleasing
- • efficient for beginners
- • efficient for advanced users
- natural for the application domain
- responsive
- scalable to demanding uses

Some UI design principles:

✱ Provide interaction using vocabulary of the application domain

✱ Direct manipulation

- WYSIWYG — What you see is what you get
- Physically obvious + intuitive pointer motion
- Labeled buttons for actions
- Immediate display of results w/ intermediate feedback
- Easily reversible commands (UNDO)

→ ✱ Modeless design (few dialog boxes, few tool modes)

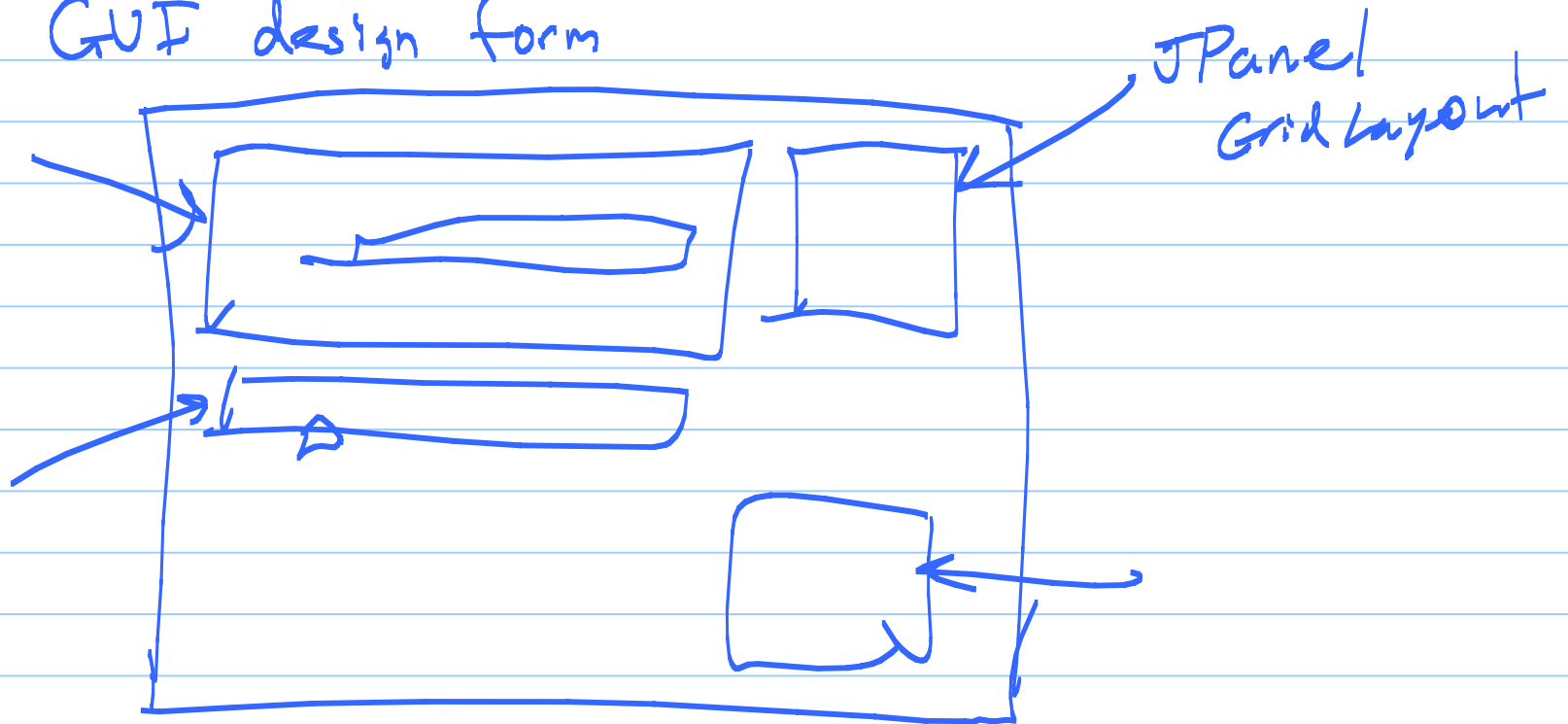
✱ Avoid clutter

✱ Consistency across multiple applications (familiarity)

① Partners

② 132 Web Site Forms

GUI design form



Critique

- criticize the work, not the person
- giving advice/suggestions may save the other person a lot of time

* I'm confused —
i'm not sure how I would ...

* Concrete suggestions —
but in-line with goals

