Design Process

Forward-thinking design — think things through

Software is soft, malleable

⇒ never finished
⇒ may be used in ways the designer never imagined

Goals:

- Flexible — reusable parts (orthogonality)
- Extensible (you can add completely new things easily)
- Simple (consistent philosophy comes through)
How? 

1. **Think of all possible uses & features & implement them all**
   $$\Rightarrow$$ never finish ... never get started!

2. **Think about the kinds of things that might be needed**
   & ensure the design can support important ones

   1. Brainstorm — a list of possible "use cases"
      (stop when gets ridiculous)

   2. Evaluate competing designs &
      see how they would extend to support the case cases

   3. Decide on the most useful set of
      features & a supporting design
Use cases covered:

Basic Framework

Implementation effort (lines of code)

Foundation

Easy extensions

Diminishing returns
Survey Brainstorming: What if the survey designer wanted to...

- add a picture to a question? ...an answer?
- change a question from "choose one" to "select all that apply?"
- create a question where subject ranks responses?
- next question depends on time/correctness?
- next question depend on entire history of responses?
- have sections of the survey that could be skipped (mini-surveys?)
- reorder answer choices?
- allow easy ext.
- present answers choices in a random order?
- present questions in random order?
- reorder the questions?
- copy a question?
sections of the survey that could be skipped (mini-surveys?)

Simple, but doesn't support

1. Survey <-> Question

2. Survey <-> Question
   - Section
   - SurveyItem
   - add(SurveyItem...)

3. Survey <-> Question