Community Design
and the KDE Visual Design Group
My Background

● Avionics Safety Engineer by day
● Bangarang developer
● Lifelong fan of all design
  ○ Cars
  ○ Architecture
  ○ Hardware and software
What is Community Design?

- Design in the open
- Everyone welcome to participate regardless of skill.
- Positive environment where learning is encouraged and enabled.
Why?

- Design needs to be liberated from its ivory tower and lone-genius myths.
- KDE especially, is driven and sustained by community. KDE *is* community.
- Establish design a sustainable, core competency in KDE.
Challenges

- “Design-by-committee” fears
- Building and sustaining a community
- Learning how to best interaction with developers
“Design by Committee” fears

- Myth - More people necessarily results compromised or bad design.
- Reality - More people *can* result in compromised or bad design.
- Reality - More people *also* means more ideas, more creativity, more ways to discover good design solutions.
“Design by Committee” fears

- Similar arguments were made about FLOSS development.
- We can and have learned lessons from the FLOSS developer story.
  - Same skepticism - “a bunch of people working what they want for free can’t build high quality software”...
  - Well, we know how that turned out...
“Design by Committee” fears

- So how are these fears addressed?
  - Designers organically curate design feedback on their proposed design.
  - Designers who have earned influence chime in on proposed designs.
  - Limit time to produce the result.
  - Not much different from the FLOSS developer story.
Building and Sustaining a Community

● Encouraging a learning environment
  ○ Explicitly encourage constructive feedback
  ○ Set an example
  ○ Draw the line on destructive behavior

● Providing tools
  ○ Forums (forum.kde.org), including “quiet area”
  ○ Visual Design Guidelines (integrated with KDE HIG)
  ○ Mockup Toolkit
Building and Sustaining a Community

- Multiple UI design disciplines
- Dreams fulfilled are built on record of success
  - Never turn down an opportunity to be effective (correctness * commitment)
  - Know our deliverable - design
  - Talk about what comes next
- Adjust on the fly
Interaction with developers

Developer requests help from Design Community. Initially work together on design needs. Organically identify a Resident Designer.

Design Community

Developer

Resident Designer
Includes visual or usability design disciplines

Work together on day-to-day design issues:
- Review requests
- Design-related bugs

Work together to brainstorm design solutions for project and bug triage for larger projects.
The Community Design Cycle

1. Develop a candidate design
2. Announce cycle including design and target length of cycle
3. Incorporate design feedback into candidate design.
4. Post design update.
5. Repeat 3 and 4 till cycle time expires.
6. Announce end of cycle.

Lesson learned: Focus on constructive design feedback, ignore destructive feedback.
Community design so far...

- Cursor theme
- Krita toolbar icons
- Breeze UI controls design
- Breeze window decoration design
- System Settings design
- Plasmoid designs
- Plasma Media Center design
Community design so far...

- Not everything the VDG has produced emerged from community design.
  - Things like the plasma theme, the main icon set, the login/logout/lockscreen design.
  - But, over time, more designs are coming from the public forums.
Community design so far...

- We’re learning
  - More use-case/requirements capture and communication to support candidate designs.
  - More effort to build a broad-based community understanding of positive designer interaction with developers.
  - Improve consistent communication of design specs to developers.
The road ahead

Design Guidelines
- Design Vision and Principles
- Style guidance
- Integrated with HIG
- Mockup toolkit

Design Patterns
- Application layout patterns
- Functional patterns
- Search pattern
- Date/Time picker
- Color picker
- Alert and Info patterns
- more

Complete Application Designs
- Smaller applications first
- Larger, more complex application with experience

Complete Desktop/OS Design
- Conceptual OS user experience design
- No concern about underlying technology
- Just design the entire user experience
- Assume the underlying technology stack will be sufficient

The order is priority not necessarily sequencing
Ignore the priority if the opportunity arises to be effective (correctness x commitment).
Visual Design and QML

- Workshop/Tutorial tomorrow @ 14:00 Room 2
  - No QML experience required
Community Design and the KDE VDG

Questions?