Usability Goals

Speed:

The system should perform its actions very quickly, with almost imperceptible delays for simple actions like searching, querying the database, and accessing help and documentation.

Questions

- Is there only a short or nonexistent delay for most actions?
- How long does a user have to wait for the system to perform operations?

Learnability:

The system should be self-documenting, with help messages, tooltips, and useful information embedded into the interface rather than exclusively in external documentation. When there is a problem (some form of error, or otherwise), the system should provide informative and helpful messages that direct the user towards solutions. In addition to in-system help, there should be documentation that clearly and thoroughly details the whole system, how to use it, and all functionality.

Questions

- Is the documentation high quality (thorough, easy-to-read, well laid out, etc.)
- Are functions documented in-place, in the documentation, or both?
- If a user wants information on a specific operation, how quickly can they find it?
- Does the system have easy to understand concepts, metaphors, etc.?
- Does the system have tutorials?
  - Do they cover a large range of functionality?
  - How much of the system can users learn through tutorials?
  - How easy are the tutorials to follow?

Memorability:

The control buttons should be designed and constructed in such a way as to indicate in which order to use them, and which ones are valid to be used at different times. The operations which are most relevant to the current situation should be front and center in the most obvious position and designed in a way that makes them stand out. Operations that are currently invalid should be faded out, or simply not presented at all.

Keyboard shortcuts should be mnemonic if possible, or have some other way to help with memorization. Users should be able to easily see common keyboard shortcuts at a glance.

Questions

- Can users easily remember how to do certain operations?
- Can users easily remember where to find particular functions?
- Can users easily remember keyboard shortcuts?
- Does the system guide users in a way that encourages memorization?

Utility:

The system should provide the necessary set of functionality to do package management. The basic functions, like adding, removing, and updating packages, and more advanced functions.

Questions

- Does the system provide the necessary functionality to manage the software installed on the user's device.

Safety:

The UI should be designed in such a way that it makes it hard for users to do actions that permanently and irreversably lose work. One way to do this is with an "Undo" function, others are through thoughtful placement of commands, confirmation prompts, and making the system informative enough for the users to be sure that the commands they are issuing do what they expect it to.

The physical aspect of safety (saftey from bodily harm) is not revalent to this system.

Questions

- Does the UI help prevent users from making mistakes?
- Can any changes the user makes be undone?
- Does the system inform the users of what each function does?
  - Do the users understand this information?

Efficiency:

How well does the system support the user in performing their tasks. Do simple actions require multiple steps, or very few steps?

Questions

- Can a user perform the basic actions of the system in a very small number of steps.

User Experience Goals

Positive: Ones We Want to Create

- Helpful: When using the system, how much help do users receive from it in completing their tasks?
-

Negative: Ones We Want to Prevent

- Frustrating: Does the user have to struggle with the system to get what they need done? Does it get in their way and limit them?
- Confusing: Does the user understand how to use the system's functionality? Does it make sense? Do user's often find themselves confused about what a function does, what an error message means, or where to find what they need?