Somewhat Simplified Solitaire Design Document

Date: 4 November 2011

Document Author(s): Will Blazer, Andrew Kofink

Design Rationale

1. The UI controls the interaction with the user and includes the main method. It makes a new Driver which makes a new Deck to encrypt/decrypt/etc.
2. The input to this program is a string, and the output is also a string. Only these few classes are really needed to modify the input and output it to the console. Data is all handled in Strings, chars, and ints.
3. The Driver does nearly everything to do with encryption/decryption, but it uses a Deck of cards to compute the encryption/decryption. The UI is necessary to separate the main method and interface from the action of the program itself (i.e. the brunt work of enc/dec).
4. Since nothing inherits properties or behaviors from anything else, there are no inheritance relationships. We thought about making Card a class, but it does not have enough behavior to implement as an individual class, so we just made it an instance variable of Deck. Driver uses Deck and its behavior, and UI uses Driver and its behavior.
5. The only design pattern that can be seen in our program somewhat is the model-view-controller pattern (see http://en.wikipedia.org/wiki/Model–view–controller). I say this because we have separated the very simple user interface from the very simple driving force of the program (controller, i.e. Driver).

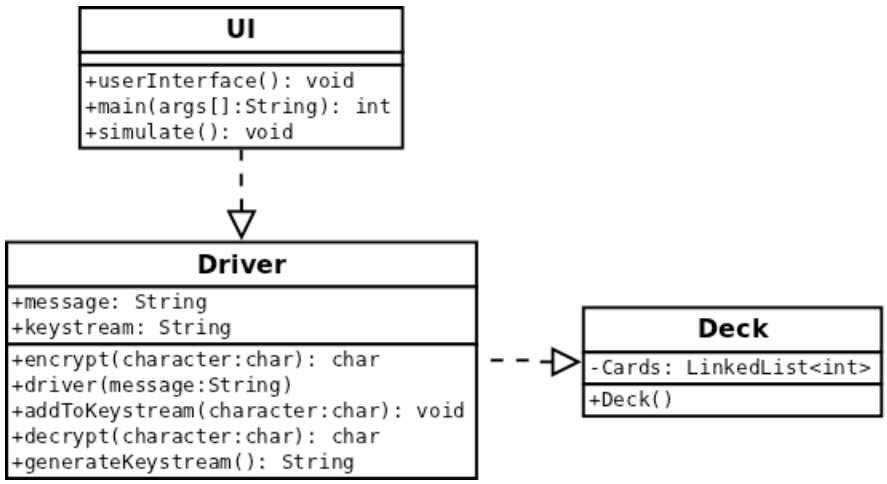

Figure 1: Class Diagram for <Project Name>

Document Revision History

Date	Author	Change Description
4 November 2011	Will Blazer, Andrew Kofink	• Iteration 0