The biggest difference is the design of the UI. A good GUI can make or break an application. Mac fans would draw attention to the beautifully designed GUI's of the average Mac OS X app and they've got a point, but this isn't a technology issue - it's a design/ethos/usability issue.

As for technical issues, in no particular order:

1. The user can do anything they want in any order at any time, unlike console program in which you're either asking for input or telling them the output. You cannot assume that they'll follow the order you hope, unless you enforce the workflow Wizard-stylee.
2. As already mentioned, events play a big part in this, and you can get multiple events happen while you're servicing the last one, so you can't really construct your state based on the 'current event'. Use closures or a similar mechanism to maintain context across different events. In a console app, your FSM is usually self-contained around the 'get input, process input, update output' loop. There isn't the same kind of structure in GUI programming - the 'main' is a re-entrant event-driven thing, often a ginormous switch() statement.
3. You need to consider different screen sizes/resolutions and allow the GUI to resize from 800x600 up to the users' monitor maximum.
4. You need to consider different input strategies - mouse, keyboard, touch, etc. Some technologies come for free (Mouse-wheel scrolling), others require some integration work (Ink).
5. Accessibility - a GUI is much more suitable for less able users who have restricted vision, hearing, motor skills or cognitive skills. A 'ding' noise is nice and obvious compared to a cryptic error message on the console.
6. Internationalization - i'm assuming your console app is US/ANSI only, but when you get into GUI, you can have language/resource packages that can target other languages and regions with no change to the coding, if you prepared for it from the start. For example, no hard-coded language strings in the code - everything as resource lookups.
7. You have lots more options for implementation technology - web-based, various GUI kits, Flash/WPF, etc.
8. Use of colour and animation. Console programs are generally monochromatic and don't animate much. Many modern GUI frameworks provide themed widgets and have move/size/show/hide animation effects, often for free.
9. Graphics. Console apps sometimes use ASCII art for diagrams, but a GUI app gives you full graphical ability. Lovely art can make a big difference too.

<https://www.ntu.edu.sg/home/ehchua/programming/java/J4a_GUI.html>