```
from time import sleep
from rich.console import Console

console = Console()
with console.screen():
    console.print(locals())
    sleep(5)
```

The above code will display a pretty printed dictionary on the alternate screen before returning to the command prompt after 5 seconds.

You can also provide a renderable to *screen()* which will be displayed in the alternate screen when you call `update()`.

Here's an example:

```
from time import sleep

from rich.console import Console
from rich.align import Align
from rich.text import Text
from rich.panel import Panel

console = Console()

with console.screen(style="bold white on red") as screen:
    for count in range(5, 0, -1):
        text = Align.center(
            Text.from_markup(f"[blink]Don't Panic![/blink]\n{count}", justify="center"),
            vertical="middle",
        )
        screen.update(Panel(text))
        sleep(1)
```

Updating the screen with a renderable allows Rich to crop the contents to fit the screen without scrolling.

For a more powerful way of building full screen interfaces with Rich, see *Live Display*.

Note

If you ever find yourself stuck in alternate mode after exiting Python code, type `reset` in the terminal

2.21 Terminal detection

If Rich detects that it is not writing to a terminal it will strip control codes from the output. If you want to write control codes to a regular file then set `force_terminal=True` on the constructor.

Letting Rich auto-detect terminals is useful as it will write plain text when you pipe output to a file or other application.

2.22 Interactive mode

Rich will remove animations such as progress bars and status indicators when not writing to a terminal as you probably don't want to write these out to a text file (for example). You can override this behavior by setting the `force_interactive` argument on the constructor. Set it to `True` to enable animations or `False` to disable them.