```
from rich.console import Console
console = Console()
with console.capture() as capture:
    console.print("[bold red]Hello[/] World")
str_output = capture.get()
```

An alternative way of capturing output is to set the Console file to a `io.StringIO`. This is the recommended method if you are testing console output in unit tests. Here's an example:

```
from io import StringIO
from rich.console import Console
console = Console(file=StringIO())
console.print("[bold red]Hello[/] World")
str_output = console.file.getvalue()
```

2.19 Paging

If you have some long output to present to the user you can use a *pager* to display it. A pager is typically an application on your operating system which will at least support pressing a key to scroll, but will often support scrolling up and down through the text and other features.

You can page output from a Console by calling *pager()* which returns a context manager. When the pager exits, anything that was printed will be sent to the pager. Here's an example:

```
from rich.__main__ import make_test_card
from rich.console import Console

console = Console()
with console.pager():
    console.print(make_test_card())
```

Since the default pager on most platforms don't support color, Rich will strip color from the output. If you know that your pager supports color, you can set `styles=True` when calling the *pager()* method.

Note

Rich will look at `MANPAGER` then the `PAGER` environment variables (`MANPAGER` takes priority) to get the pager command. On Linux and macOS you can set one of these to `less -r` to enable paging with ANSI styles.

2.20 Alternate screen

Warning

This feature is currently experimental. You might want to wait before using it in production.

Terminals support an 'alternate screen' mode which is separate from the regular terminal and allows for full-screen applications that leave your stream of input and commands intact. Rich supports this mode via the *set_alt_screen()* method, although it is recommended that you use *screen()* which returns a context manager that disables alternate mode on exit.

Here's an example of an alternate screen: