

**begin\_capture()**

Begin capturing console output. Call `end_capture()` to exit capture mode and return output.

**Return type**

None

**bell()**

Play a ‘bell’ sound (if supported by the terminal).

**Return type**

None

**capture()**

A context manager to *capture* the result of `print()` or `log()` in a string, rather than writing it to the console.

**Example**

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

**Returns**

Context manager with disables writing to the terminal.

**Return type**

`Capture`

**clear(home=True)**

Clear the screen.

**Parameters**

`home (bool, optional)` – Also move the cursor to ‘home’ position. Defaults to True.

**Return type**

None

**clear\_live()**

Clear the Live instance. Used by the Live context manager (no need to call directly).

**Return type**

None

**property color\_system: str | None**

Get color system string.

**Returns**

“standard”, “256” or “truecolor”.

**Return type**

Optional[str]

**control(\*control)**

Insert non-printing control codes.

**Parameters**

- `control_codes (str)` – Control codes, such as those that may move the cursor.