(continued from previous page)

```
        yield f"[b]Student:[/b] #{self.id}"
        my_table = Table("Attribute", "Value")
        my_table.add_row("name", self.name)
        my_table.add_row("age", str(self.age))
        yield my_table
```

If you were to print a `Student` instance, it would render a simple table to the terminal.

22.2.1 Low Level Render

For complete control over how a custom object is rendered to the terminal, you can yield *Segment* objects. A Segment consists of a piece of text and an optional Style. The following example writes multi-colored text when rendering a `MyObject` instance:

```
class MyObject:
    def __rich_console__(self, console: Console, options: ConsoleOptions) ->␣
↪RenderResult:
        yield Segment("My", Style(color="magenta"))
        yield Segment("Object", Style(color="green"))
        yield Segment("()", Style(color="cyan"))
```

22.2.2 Measuring Renderables

Sometimes Rich needs to know how many characters an object will take up when rendering. The *Table* class, for instance, will use this information to calculate the optimal dimensions for the columns. If you aren't using one of the renderable objects in the Rich module, you will need to supply a `__rich_measure__` method which accepts a *Console* and *ConsoleOptions* and returns a *Measurement* object. The Measurement object should contain the *minimum* and *maximum* number of characters required to render.

For example, if we are rendering a chess board, it would require a minimum of 8 characters to render. The maximum can be left as the maximum available width (assuming a centered board):

```
class ChessBoard:
    def __rich_measure__(self, console: Console, options: ConsoleOptions) -> Measurement:
        return Measurement(8, options.max_width)
```