CHAPTER

TWENTYTWO

CONSOLE PROTOCOL

Rich supports a simple protocol to add rich formatting capabilities to custom objects, so you can *print()* your object with color, styles and formatting.

Use this for presentation or to display additional debugging information that might be hard to parse from a typical `__repr__` string.

22.1 Console Customization

The easiest way to customize console output for your object is to implement a `__rich__` method. This method accepts no arguments, and should return an object that Rich knows how to render, such as a *Text* or *Table*. If you return a plain string it will be rendered as *Console Markup*. Here's an example:

```
class MyObject:
    def __rich__(self) -> str:
        return "[bold cyan]MyObject()"
```

If you were to print or log an instance of `MyObject` it would render as `MyObject()` in bold cyan. Naturally, you would want to put this to better use, perhaps by adding specialized syntax highlighting.

22.2 Console Render

The `__rich__` method is limited to a single renderable object. For more advanced rendering, add a `__rich_console__` method to your class.

The `__rich_console__` method should accept a *Console* and a *ConsoleOptions* instance. It should return an iterable of other renderable objects. Although that means it *could* return a container such as a list, it generally easier implemented by using the `yield` statement (making the method a generator).

Here's an example of a `__rich_console__` method:

```
from dataclasses import dataclass
from rich.console import Console, ConsoleOptions, RenderResult
from rich.table import Table

@dataclass
class Student:
    id: int
    name: str
    age: int
    def __rich_console__(self, console: Console, options: ConsoleOptions) ->␣
↪RenderResult:
```

(continues on next page)