CHAPTER

NINETEEN

TREE

Rich has a *Tree* class which can generate a tree view in the terminal. A tree view is a great way of presenting the contents of a filesystem or any other hierarchical data. Each branch of the tree can have a label which may be text or any other Rich renderable.

Run the following command to see a demonstration of a Rich tree:

```
python -m rich.tree
```

The following code creates and prints a tree with a simple text label:

```
from rich.tree import Tree
from rich import print

tree = Tree("Rich Tree")
print(tree)
```

With only a single `Tree` instance this will output nothing more than the text "Rich Tree". Things get more interesting when we call *add()* to add more branches to the Tree. The following code adds two more branches:

```
tree.add("foo")
tree.add("bar")
print(tree)
```

The tree will now have two branches connected to the original tree with guide lines.

When you call *add()* a new Tree instance is returned. You can use this instance to add more branches to, and build up a more complex tree. Let's add a few more levels to the tree:

```
baz_tree = tree.add("baz")
baz_tree.add("[red]Red").add("[green]Green").add("[blue]Blue")
print(tree)
```

19.1 Tree Styles

The Tree constructor and *add()* method allows you to specify a `style` argument which sets a style for the entire branch, and `guide_style` which sets the style for the guide lines. These styles are inherited by the branches and will apply to any sub-trees as well.

If you set `guide_style` to bold, Rich will select the thicker variations of unicode line characters. Similarly, if you select the "underline2" style you will get double line style of unicode characters.