```
import time
from typing import Any, Callable, Dict, List

from rich.console import Console
from rich.live import Live
from rich.panel import Panel
from rich.progress import Progress, SpinnerColumn, TextColumn
from rich.table import Table
from rich.text import Text


class Formatter:
    """
    A class for formatting and printing rich text to the console.
    """

    def __init__(self):
        """
        Initializes the Formatter with a Rich Console instance.
        """
        self.console = Console()

    def print_panel(
        self, content: str, title: str = "", style: str = "bold blue"
    ) -> None:
        """
    Prints a rich panel to the console with a random color.

    Args:
        content (str): The content of the panel.
        title (str, optional): The title of the panel. Defaults to "".
        style (str, optional): The style of the panel. Defaults to "bold blue".
    """
    import random

    colors = [
        "red",
        "green",
        "blue",
        "yellow",
        "magenta",
        "cyan",
        "white",
    ]
    random_color = random.choice(colors)
    panel = Panel(
        content, title=title, style=f"bold {random_color}"
    )
    self.console.print(panel)

def print_table(
    self, title: str, data: Dict[str, List[str]]
    ) -> None:
        """
        Prints a rich table to the console.

        Args:
            title (str): The title of the table.
            data (Dict[str, List[str]]): A dictionary where keys are categories and values are lists of
capabilities.
        """
        table = Table(show_header=True, header_style="bold magenta")
        table.add_column("Category", style="cyan")
        table.add_column("Capabilities", style="green")

        for category, items in data.items():
            table.add_row(category, "\n".join(items))

        self.console.print(f"\n {title}:", style="bold yellow")
        self.console.print(table)

    def print_progress(
        self,
        description: str,
        task_fn: Callable,
        *args: Any,
        **kwargs: Any,
    ) -> Any:
```

```
    """
    Prints a progress bar to the console and executes a task function.

    Args:
        description (str): The description of the task.
        task_fn (Callable): The function to execute.
        *args (Any): Arguments to pass to the task function.
        **kwargs (Any): Keyword arguments to pass to the task function.

    Returns:
        Any: The result of the task function.
    """
    with Progress(
        SpinnerColumn(),
        TextColumn("[progress.description]{task.description}"),
    ) as progress:
        task = progress.add_task(description, total=None)
        result = task_fn(*args, **kwargs)
        progress.update(task, completed=True)
    return result

def print_panel_token_by_token(
    self,
    tokens: str,
    title: str = "Output",
    style: str = "bold cyan",
    delay: float = 0.01,
    by_word: bool = False,
) -> None:
    """
    Prints a string in real-time, token by token (character or word) inside a Rich panel.

    Args:
        tokens (str): The string to display in real-time.
        title (str): Title of the panel.
        style (str): Style for the text inside the panel.
        delay (float): Delay in seconds between displaying each token.
        by_word (bool): If True, display by words; otherwise, display by characters.
    """
    text = Text(style=style)

    # Split tokens into characters or words
    token_list = tokens.split() if by_word else tokens

    with Live(
        Panel(text, title=title, border_style=style),
        console=self.console,
        refresh_per_second=10,
    ) as live:
        for token in token_list:
            text.append(token + (" " if by_word else ""))
            live.update(
            Panel(text, title=title, border_style=style)
        )
        time.sleep(delay)



formatter = Formatter()
```